Revised: 7/20/2024

Memory Architecture I

Michael C. Hackett

Assistant Professor, Computer Science

COMMUNITY COLLEGE OF PHILADELPHIA

Lecture Topics

- Sequential Circuits
  - Clocks
  - SR Latch
  - Flip-Flops
    - SR Flip-Flop
    - D Flip-Flop
    - JK Flip Flop
  - Registers
- Memory Hierarchy
- Memory Technologies
  - RAM
  - ROM

Sequential Circuits

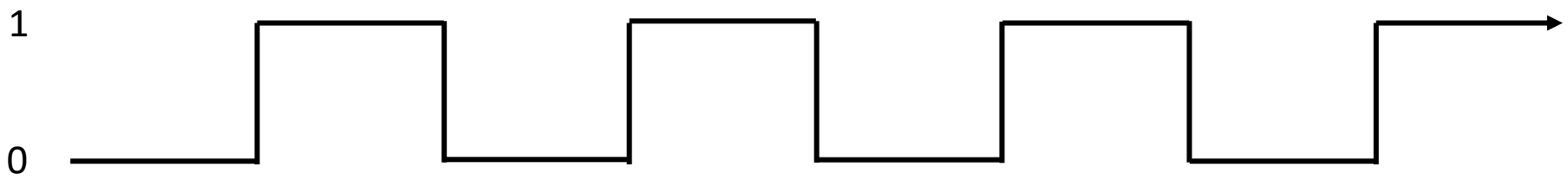
- The combinational logic circuits we've seen have the following limitations:
  - They have no memory capability
  - The output changes as soon as the input changes

- **Sequential logic circuits** maintain a state- The circuit's output is determined by both:
  - The input it currently has
  - The input it has received over time

Clocks

- In most cases, the components in a sequential circuit need to be synchronized
  - The components in the circuit must all update their states at the same time

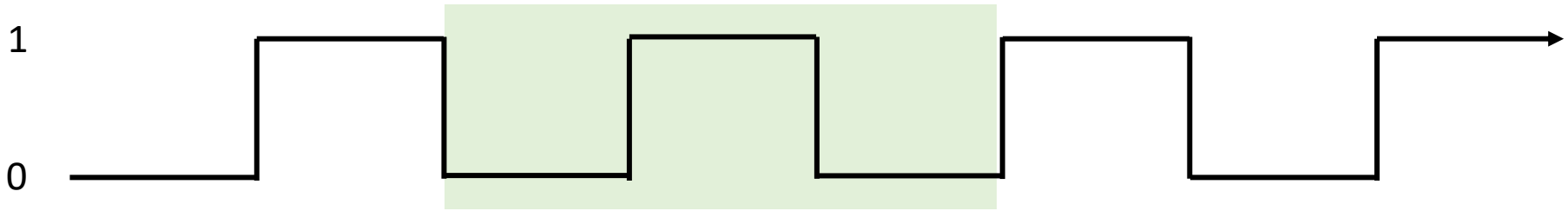
- This is achieved with a **clock signal**
  - A 1-bit signal that alternates between 1 and 0 at regular intervals



Clocks

- A **clock cycle** (“*tick*”) is when the clock transitions from 0 to 1 and back to 0

- A **clock period** is the *time* it takes to complete 1 clock cycle



Clocks

- The **clock frequency** (also called the **clock rate**) is the number of clock periods in some amount of (*wall clock*) time
  - **Wall clock time** is the seconds, minutes, hours, etc. that we experience
  - Time ticks by on a wall clock by the second, whereas these clocks may tick thousands (or millions or billions) of times per one second

- Clock frequency is measured in $\frac{cycle}{second}$ or Hertz (abbreviated Hz)

Clocks

- Clock frequency is the inverse of the clock period (and vice versa)

$$Period = \frac{seconds}{cycle} = \frac{1}{\frac{cycles}{second}} = \frac{1}{Frequency}$$

Seconds per cycle

$$Frequency = \frac{cycles}{second} = \frac{1}{\frac{seconds}{cycle}} = \frac{1}{Period}$$

Cycles per second

Clocks

- Example: What is the clock rate of a processor with a clock period of 250 ps (picoseconds)?
  - $250ps = 250 * 10^{-12}seconds = 0.00000000025s$

- $Period = \frac{250 \times 10^{-12}seconds}{1\ cycle} = \frac{0.00000000025\ seconds}{1\ cycle} = 0.00000000025\ \frac{seconds}{cycle}$

- $Frequency = \frac{1}{Period} = \frac{1}{\frac{250 \times 10^{-12}seconds}{1\ cycle}} = \frac{250 \times 10^{-12}cycles}{1\ second} = \frac{4{,}000{,}000{,}000\ cycles}{1\ second}$

$$= 4{,}000{,}000{,}000\ \frac{cycles}{second} = 4.0\ \text{GHz}$$

Clocks

- Example: What is the clock period of a processor with a clock rate of 3.8GHz?

- $Frequency = 3{,}800{,}000{,}000 \frac{cycles}{second} = \frac{3{,}800{,}000{,}000\ cycles}{1\ second}$

- $Period = \frac{1}{Frequency} = \frac{1}{\frac{3{,}800{,}000{,}000\ cycles}{1\ second}} = \frac{1\ second}{3{,}800{,}000{,}000\ cycles}$

  $= 2.63 \times 10^{-10} \frac{seconds}{cycle} = 263 \times 10^{-12} \frac{seconds}{cycle}$

  $= 263 \frac{picoseconds}{cycle}$

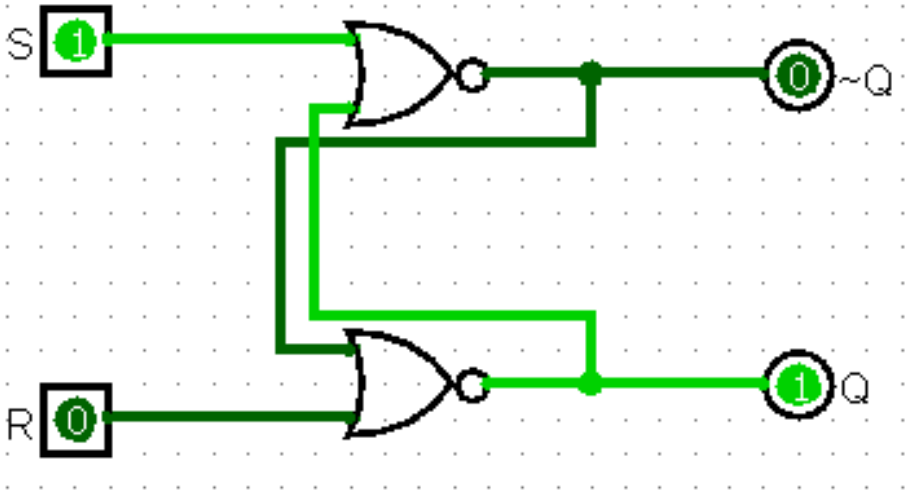
Clocks

- Abstraction of a clock:

SR Latch

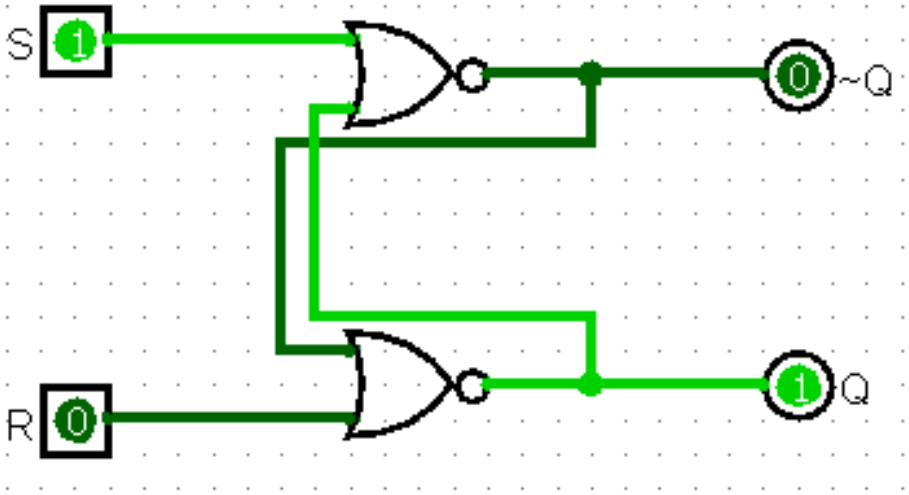
- A **latch** is a digital component that stores 1 bit of information

- An SR Latch has two inputs (Set and Reset) and two outputs (Q and its complement, $\bar{Q}$)



SR Latch

- **Feedback**
  - The output of the top NOR is one input to the bottom NOR
  - The output of the bottom NOR is one input to the top NOR



SR Latch

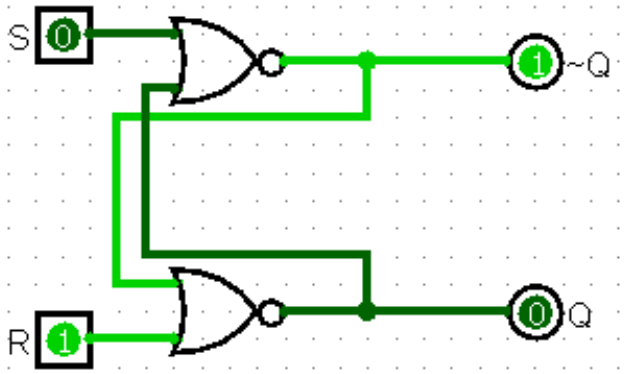
- Recall that, for a NOR gate, the output is 0 if either input is a 1

x	y	$x\ NOR\ y$
0	0	1
0	**1**	**0**
1	**0**	**0**
1	**1**	**0**

SR Latch

- We'll start with the second row of the latch's truth table:
  - This is the Reset state

S	R	$\overline{Q}$	Q	State
0	0			
0	1	1	0	Reset (Q = 0)
1	0			
1	1			

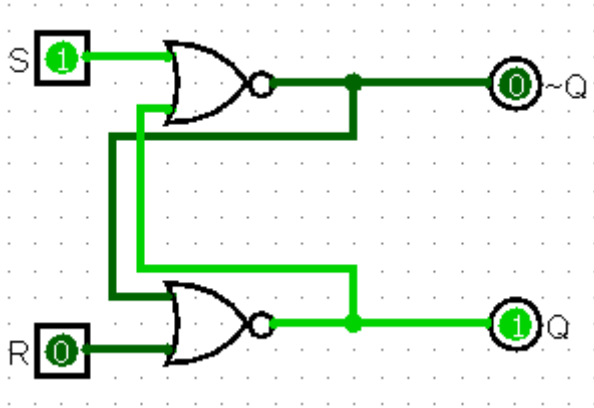


- R is 1, meaning the bottom NOR must have an output (Q) of 0
- S is 0 and the output of the bottom NOR was 0, so the top NOR must have an output ($\overline{Q}$) of 1
  - This is a valid state since Q and $\overline{Q}$ are complements

SR Latch

- Next is the third row of the latch's truth table:
  - This is the Set state

S	R	$\overline{Q}$	Q	State
0	0			
0	1	1	0	Reset (Q = 0)
1	0	0	1	Set (Q = 1)
1	1			

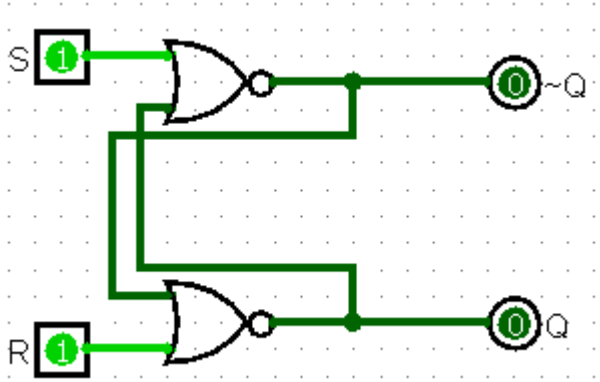


- S is 1, meaning the top NOR must have an output ($\overline{Q}$) of 0
- R is 0, and the output of the top NOR was 0, so the bottom NOR must have an output (Q) of 1
  - This is a valid state since Q and $\overline{Q}$ are complements

SR Latch

- Next is the fourth row of the latch's truth table:
  - This is the Unknown state

S	R	$\bar{Q}$	Q	State
0	0			
0	1	1	0	Reset (Q = 0)
1	0	0	1	Set (Q = 1)
1	1	0	0	Unknown



- S is 1 and R is 1 meaning both NOR gates must have an output of 0
- This doesn't make sense since Q and $\bar{Q}$ are supposed to be complements

SR Latch

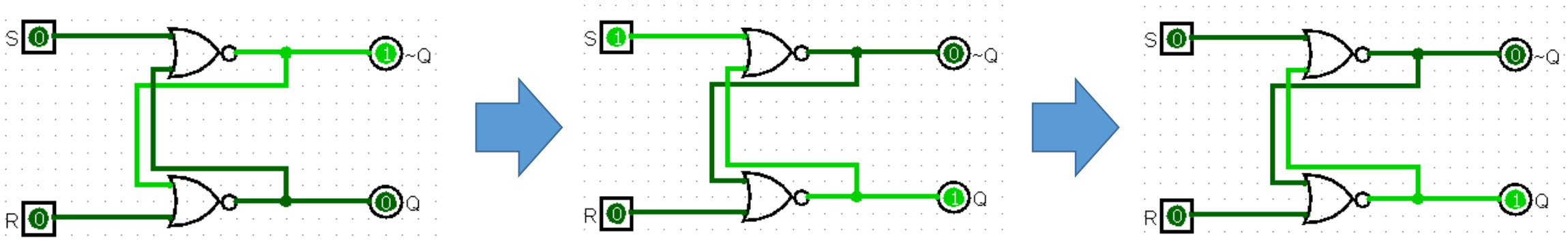
- Back to the first row of the latch's truth table:
  - This is the Unchanged state

S	R	$\bar{Q}$	Q	State
0	0	$\bar{Q}$	Q	Unchanged
0	1	1	0	Reset (Q = 0)
1	0	0	1	Set (Q = 1)
1	1	0	0	Unknown

- S is 0 and R is 0, the current state will depend on the previous state
  - If the previous state was the set state, the current state will still be in the set state
  - If the previous state was the reset state, the current state will still be in the reset state

SR Latch

- Turning Q from 0 (Reset) to 1 (Set)
  - S = 0, S = 1, S = 0

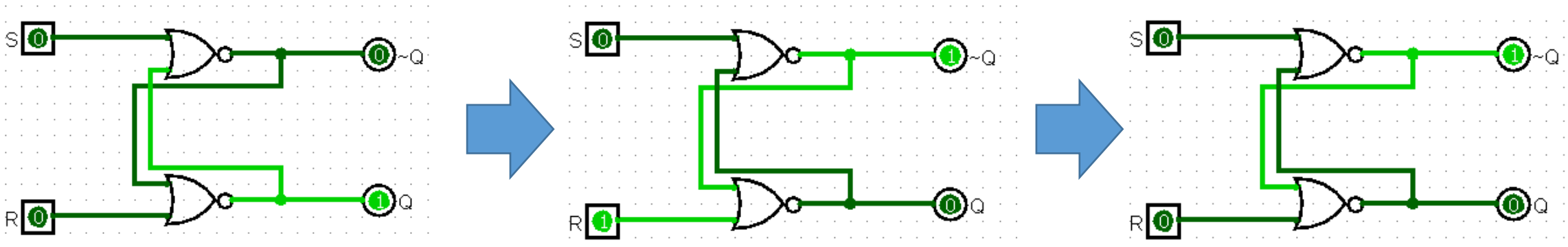


Set

Unchanged

SR Latch

- Turning Q from 1 (Set) to 0 (Reset)
  - R = 0, R = 1, R = 0

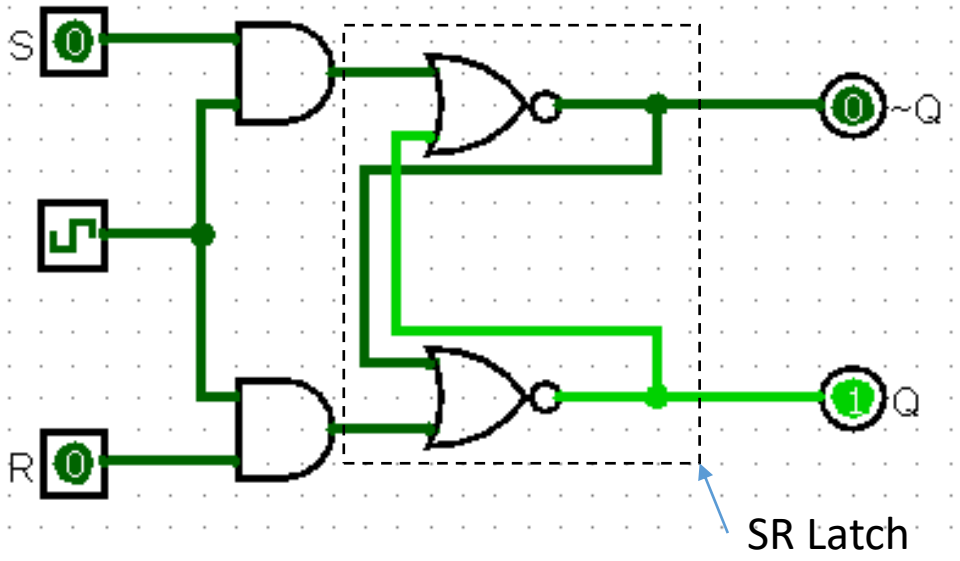


Reset

Unchanged

SR Flip-Flop

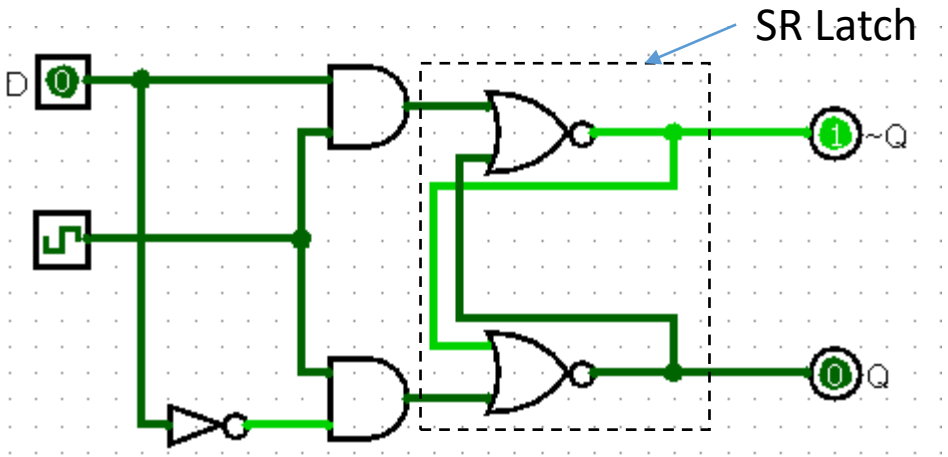
- An extension of the SR Latch is the SR Flip-Flop.
- The S and R inputs are each and'ed with a clock signal.
  - The state can only be changed when the clock signal is 1



S	R	Clock	$\bar{Q}$	Q	*State*
0	0	1	$\bar{Q}$	Q	Unchanged
0	1	1	1	0	Reset (Q = 0)
1	0	1	0	1	Set (Q = 1)
1	1	1	0	0	Unknown
X	X	0	$\bar{Q}$	Q	Unchanged

D Flip-Flop

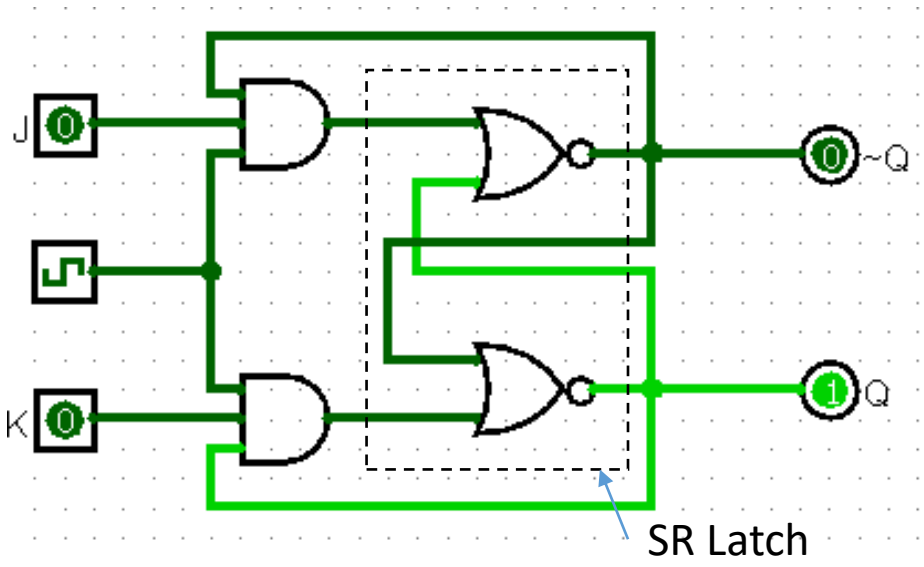
- A D Flip-Flop has one input (D) and two outputs (Q and its complement, $\bar{Q}$)
- The input is and'ed with a clock signal prior to the NOR gates.
  - The state can only be changed when the clock signal is 1



D	Clock	$\bar{Q}$	Q	*State*
0	1	1	0	Reset (Q = 0)
1	1	0	1	Set (Q = 1)
X	0	$\bar{Q}$	Q	Unchanged

JK Flip-Flop

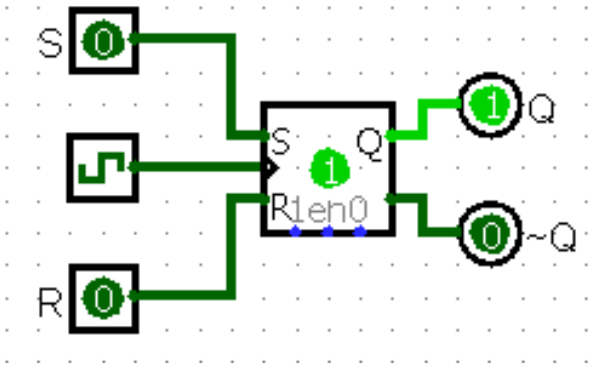
- In a JK Flip-Flop, the input is and'ed with a clock signal *and* an output prior to the NOR gates.
  - The state can only be changed when the clock signal is 1



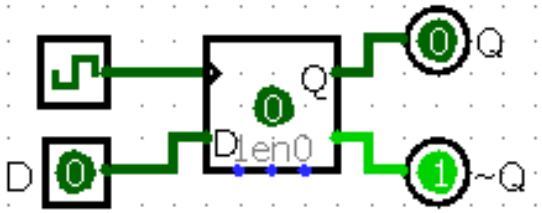
J	K	Clock	$\bar{Q}$	Q	*State*
0	0	1	$\bar{Q}$	Q	Unchanged
0	1	1	1	0	Reset (Q = 0)
1	0	1	0	1	Set (Q = 1)
1	1	1	Q	$\bar{Q}$	Toggle
X	X	0	$\bar{Q}$	Q	Unchanged

Flip-Flops

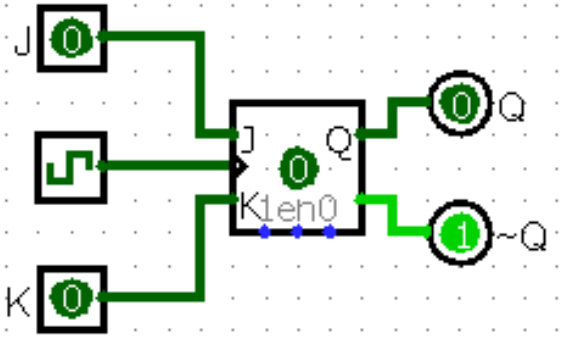
- Abstractions of Flip-Flops:



SR Flip Flop



D Flip Flop



JK Flip Flop

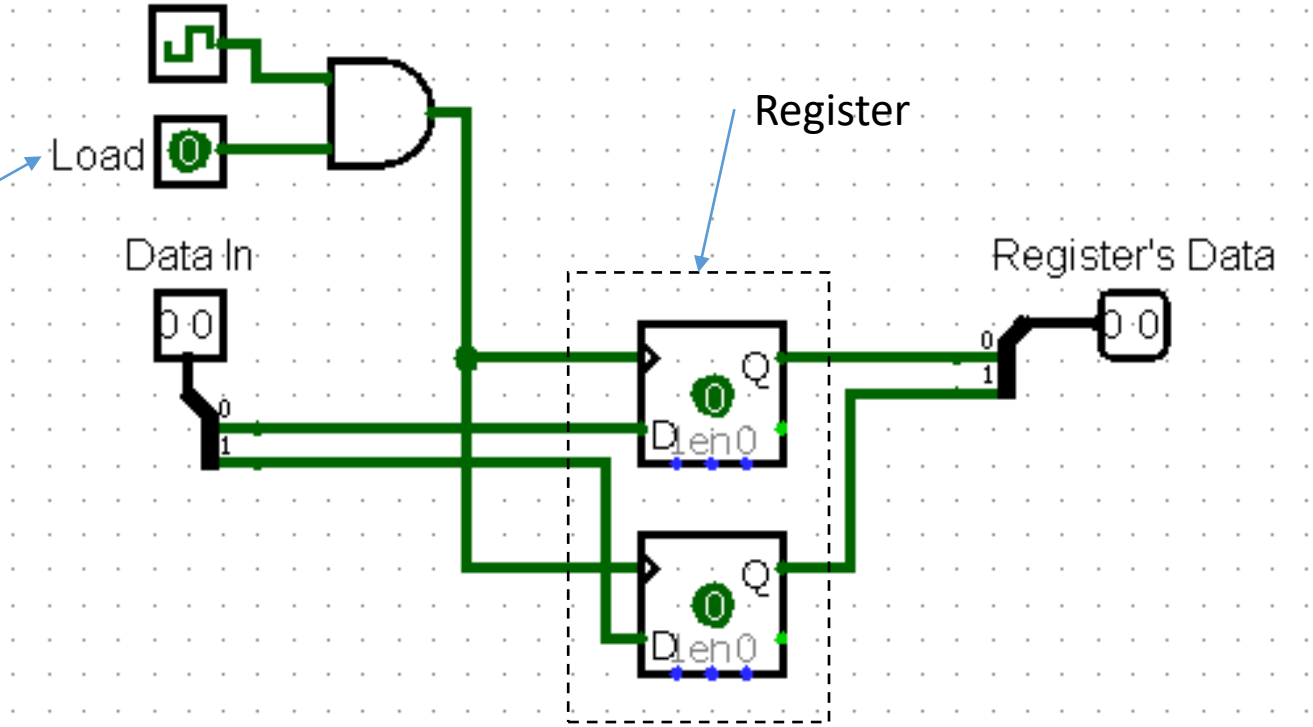
Registers

- By now, we are familiar with the use of registers from assembly programming to temporarily store data.

- Since flip-flops store 1 bit of information, we can create a register from a series of flip-flops
  - 4 flip-flops for a 4-bit register, 32 flip-flops for a 32-bit register, etc.

Registers

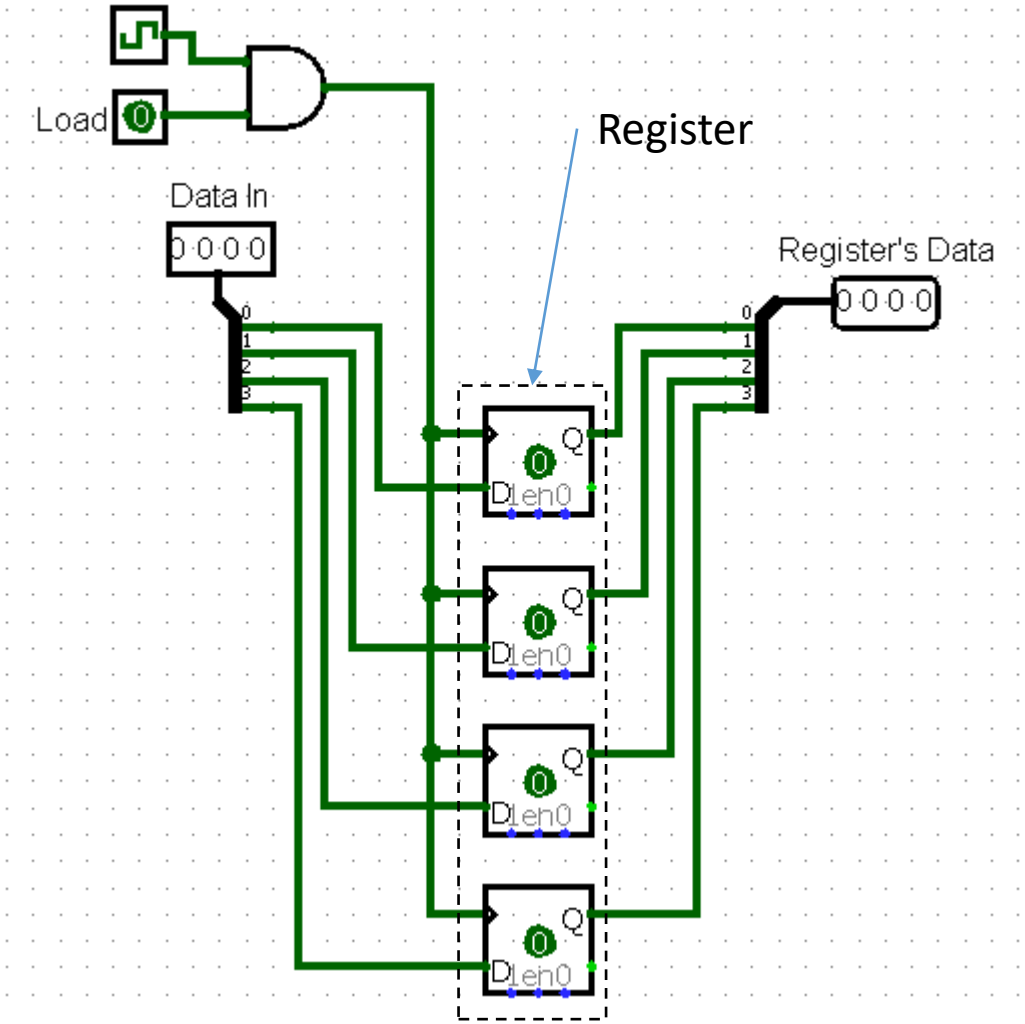
- A 2-bit register:



Enables storage of *Data In* when clock is 1

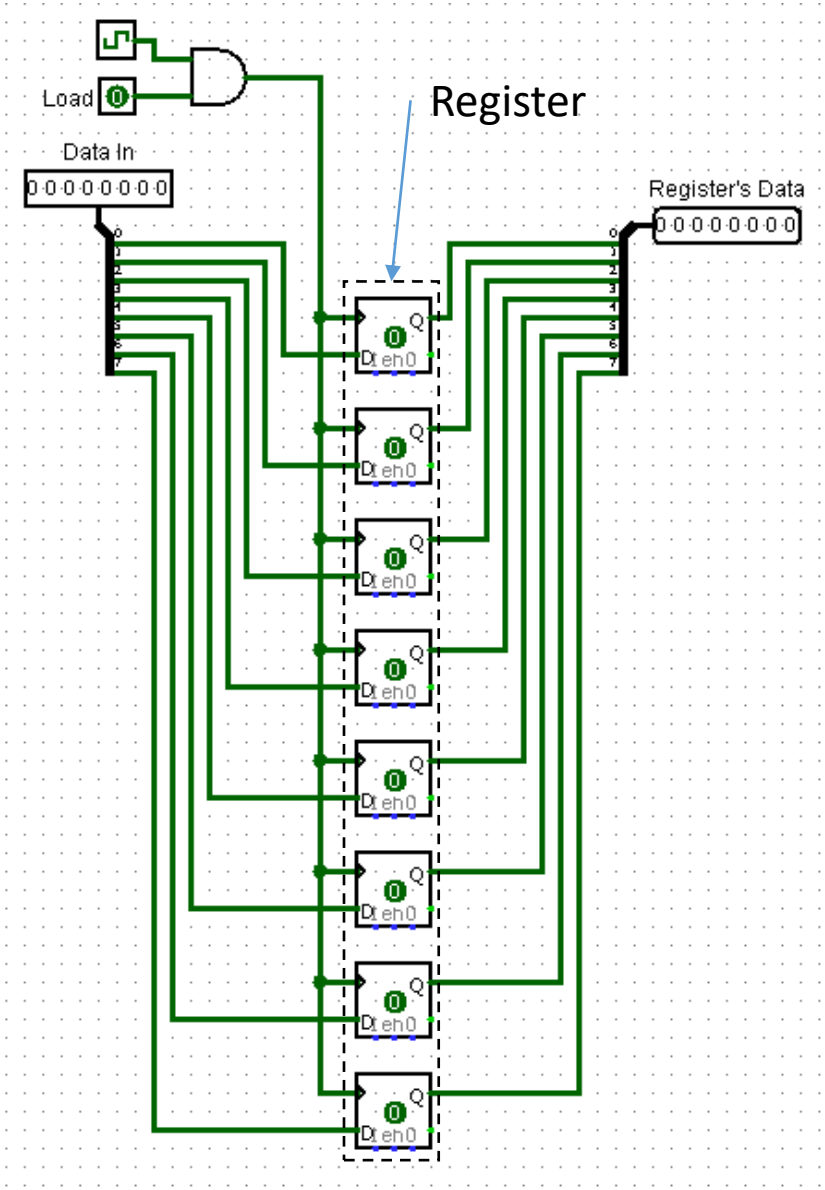
Registers

- A 4-bit register:



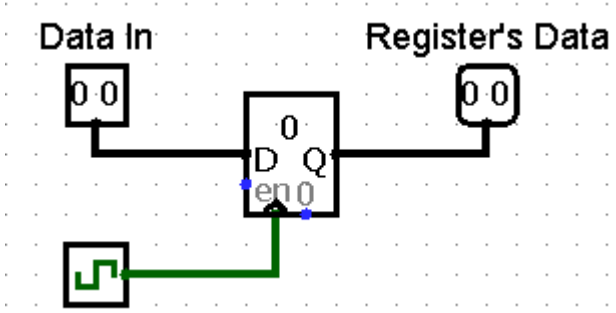
Registers

- An 8-bit register:

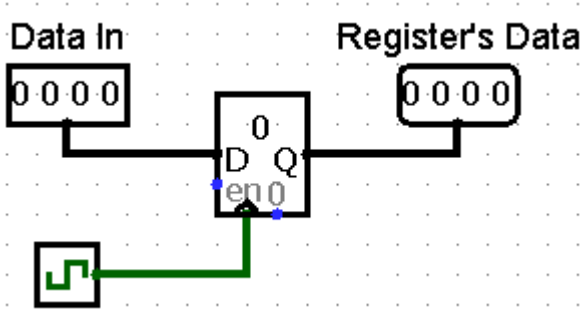


Registers

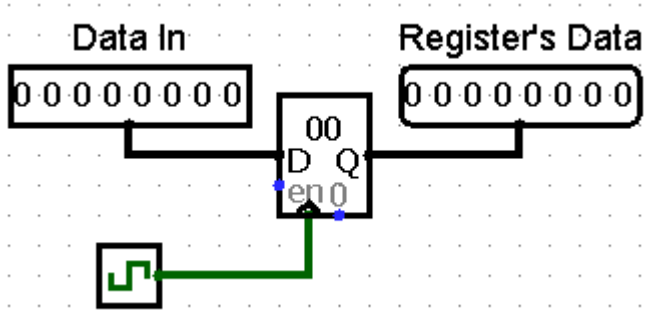
- Abstractions of Registers:



2-Bit Register



4-Bit Register



8-Bit Register

Memory Hierarchy

- The **Memory Hierarchy** is the structured levels of a computer system's memory.
  - As the distance from the processor increases, the size and the access time increases

1. Registers (Fastest/Smallest)
2. Cache Memory
3. Main Memory
4. Secondary Storage (Slowest/Largest)

Memory Hierarchy

- In the first tier are *registers*, which is memory used for storing data currently in use by the CPU.

- Registers have the fastest access time, but they are limited in number.

Memory Hierarchy

- In the second tier is *cache memory*, which is memory used for storing data the CPU has recently used.

- The cache memory is built into the CPU and has a much larger capacity than the limited number of registers.
  - Cache memory is still limited in space, as only so much can fit in the CPU.

Memory Hierarchy

- In the third tier is *main memory*, which is memory used for storing data the CPU doesn't immediately need.

- Main Memory is often referred to as the system's *RAM*
  - In this course, RAM will indicate a specific type of memory technology.

- Much larger capacity than the CPU's cache memory.

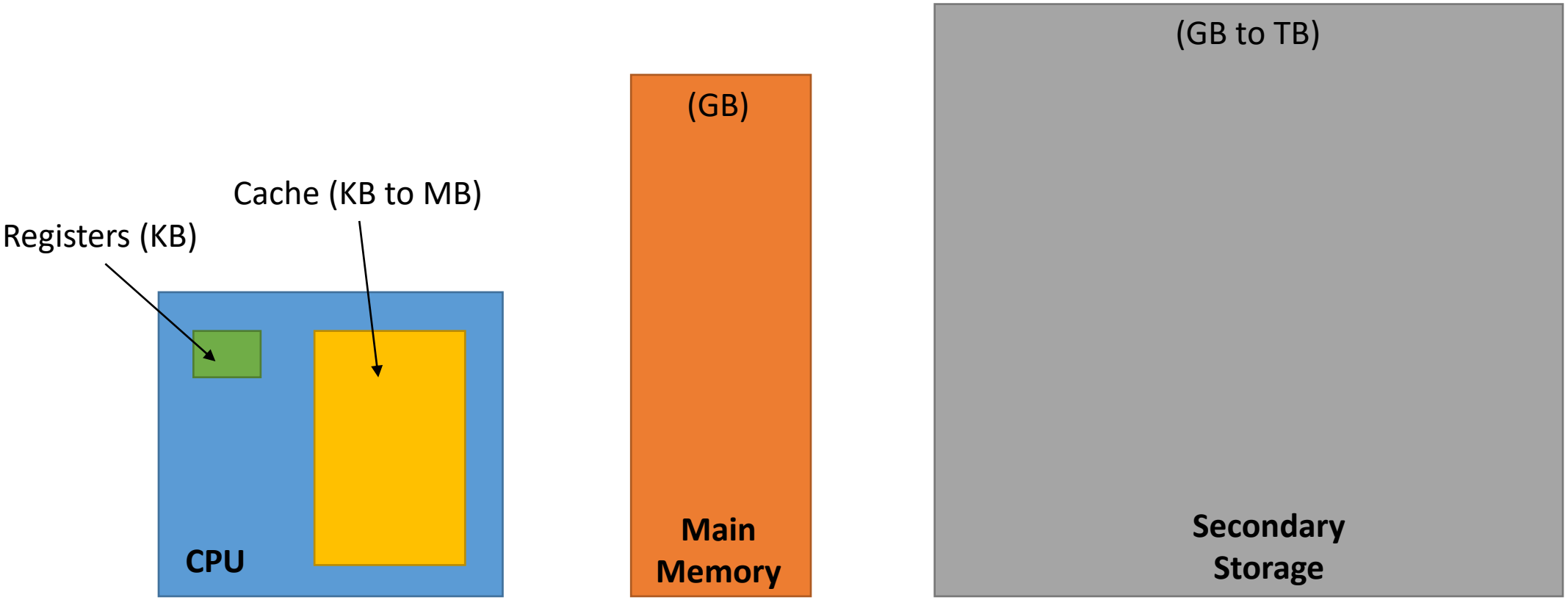
Memory Hierarchy

- In the fourth tier is *secondary storage*, which is memory used for storing data long term.
  - Magnetic Disks and Tape, Flash Memory, Optical Disks, etc.

- It is the memory with the slowest access time but has the greatest capacities.

- Secondary storage use **non-volatile** memory technologies.
  - The data stored in these technologies remains stored even when the system's power is turned off.
- Registers, cache and main memory use **volatile** memory technologies.
  - The data stored in these technologies are lost when the system's power is turned off.

Memory Hierarchy



Memory Technologies

- There are a variety of memory technologies used for secondary storage.
  - We'll discuss them in the next lecture

- This lecture focuses on the more fundamental types of memory technologies: RAM and ROM

Memory Technologies

- In older computer systems, reels of magnetic tape were used for main memory and secondary storage
- To access data, the reels were spun forward and reverse
  - *Sequential Access*

- And yes, tape is still a thing for secondary storage
  - Cost efficient means of backing up data

RAM

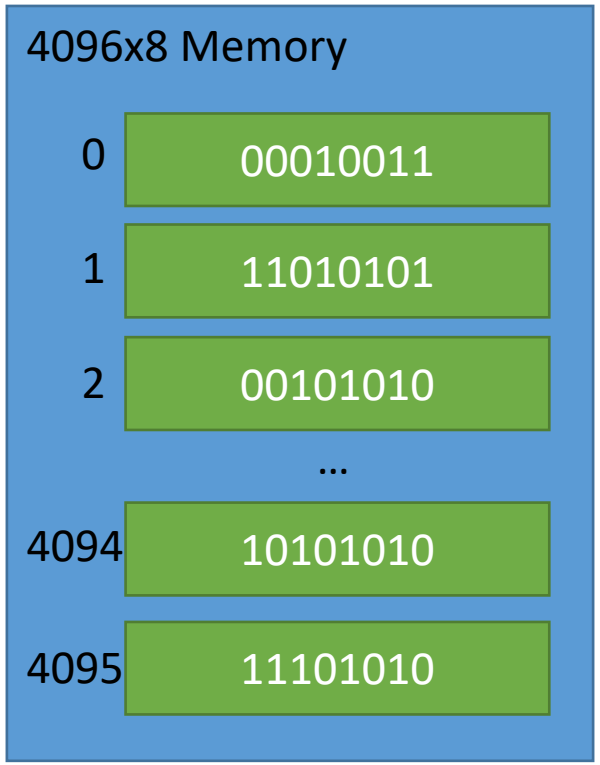
- **RAM** (**R**andom **A**ccess **M**emory) is a type of volatile memory that does not retain its stored bits when the system is powered off.

- Data stored in RAM can be accessed at random using an address

- Consists of N words of M bits each ($\boldsymbol{N \times M}$ **memory**)
  - Each word has a unique address

- For example, a 4096×8 memory:
  - 4096 8-bit words (32768 bits total)
  - 4096 unique addresses (0 through 4095)

RAM


4096x8 Memory
0
00010011
1
11010101
2
00101010
...
4094
10101010
4095
11101010

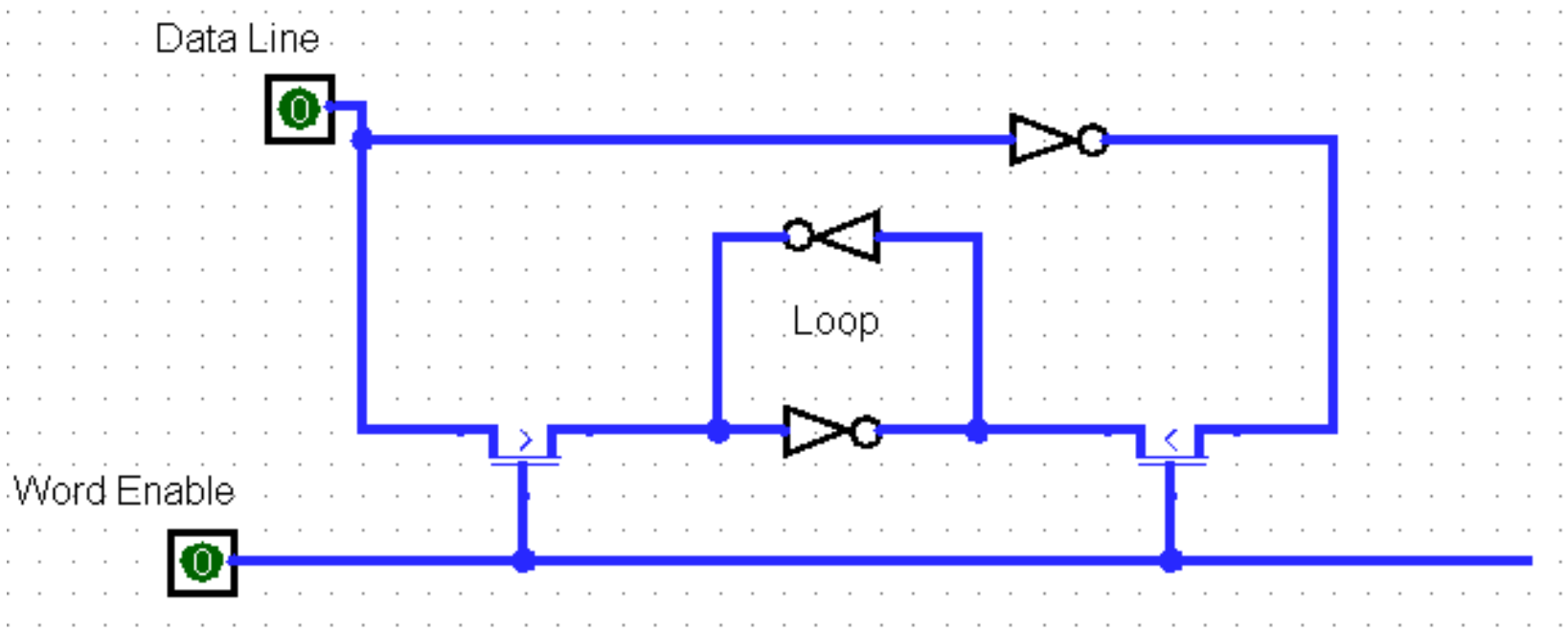
RAM

- **S**tatic **R**andom **A**ccess **M**emory (**SRAM**) and **D**ynamic **R**andom **A**ccess **M**emory (**DRAM**) are types of RAM built from an array of memory cells

- Each cell can store one bit of information
  - SRAM memory cell: Uses transistors and a loop of not gates
  - DRAM memory cell: Uses transistors and a capacitor

- Each cell is connected to a word enable line and data line
  - The **word enable line** allows the cell's value to be changed
  - The **data line** is the data to be stored in the cell

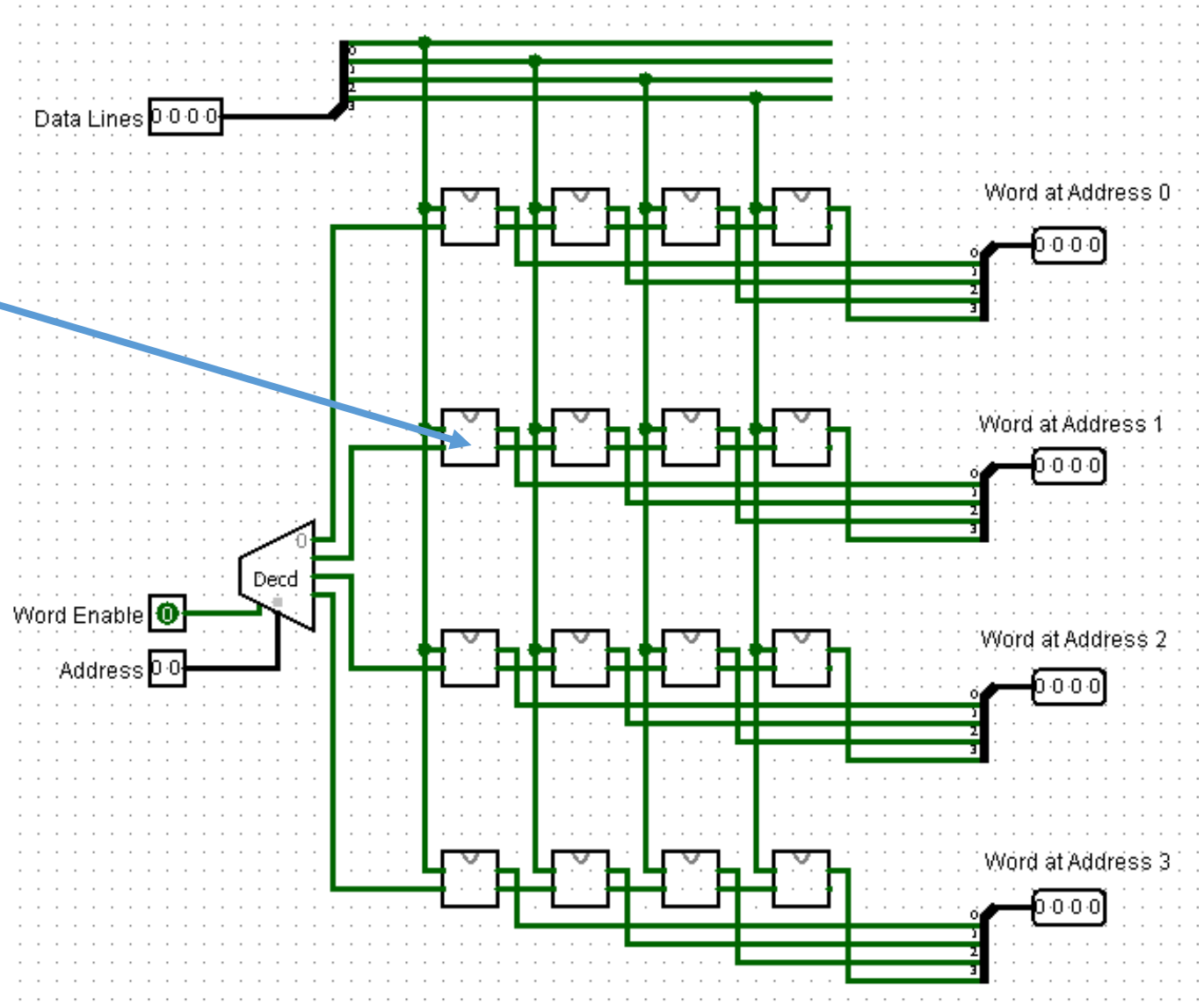
RAM – SRAM Cell



SRAM Cell

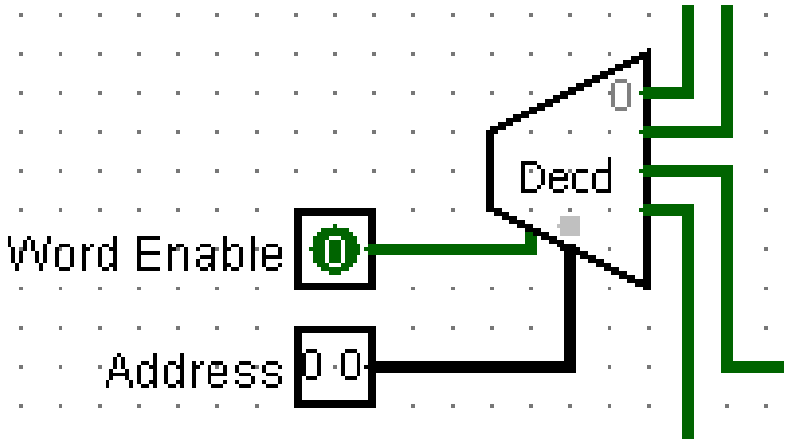
RAM

Memory Cells

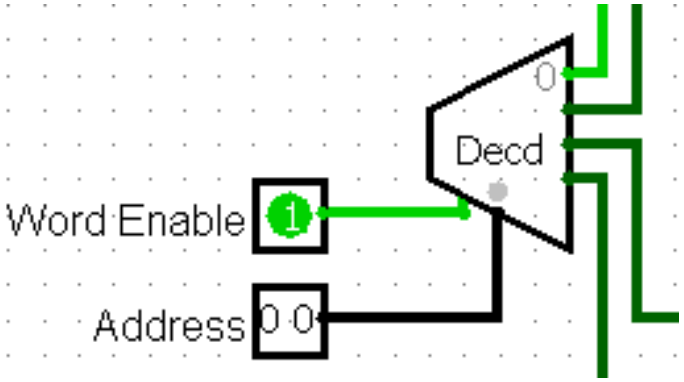


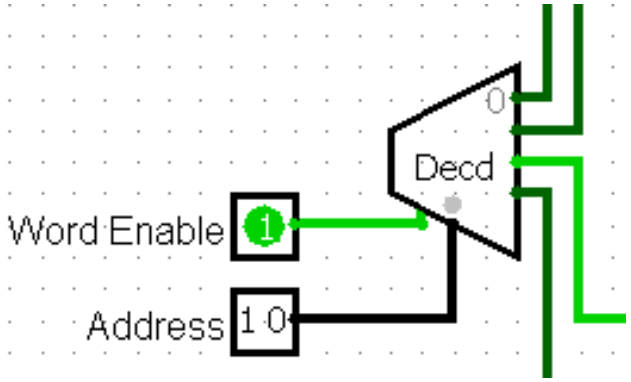
RAM

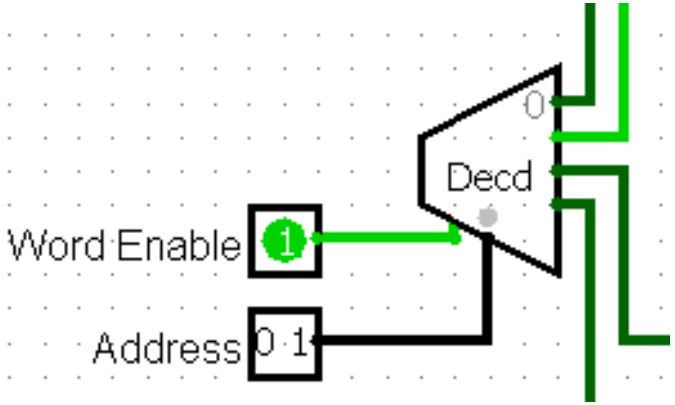
- To select a word, the word's address is decoded to enable that row of cells
  - The "Word Enable" here is controlling if the decoder is enabled
    - Each output is a word enable to a row of cells
  - The "Address" is the input to the decoder

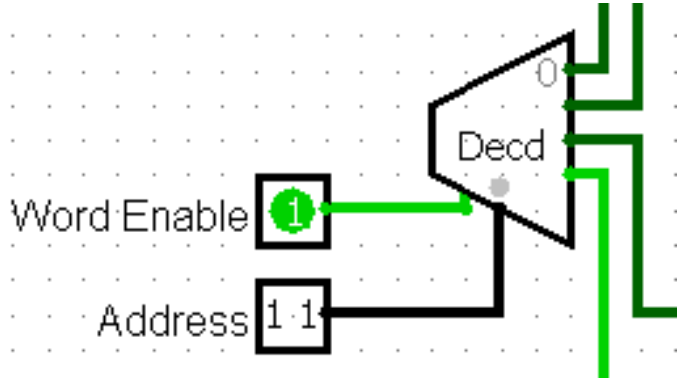


RAM

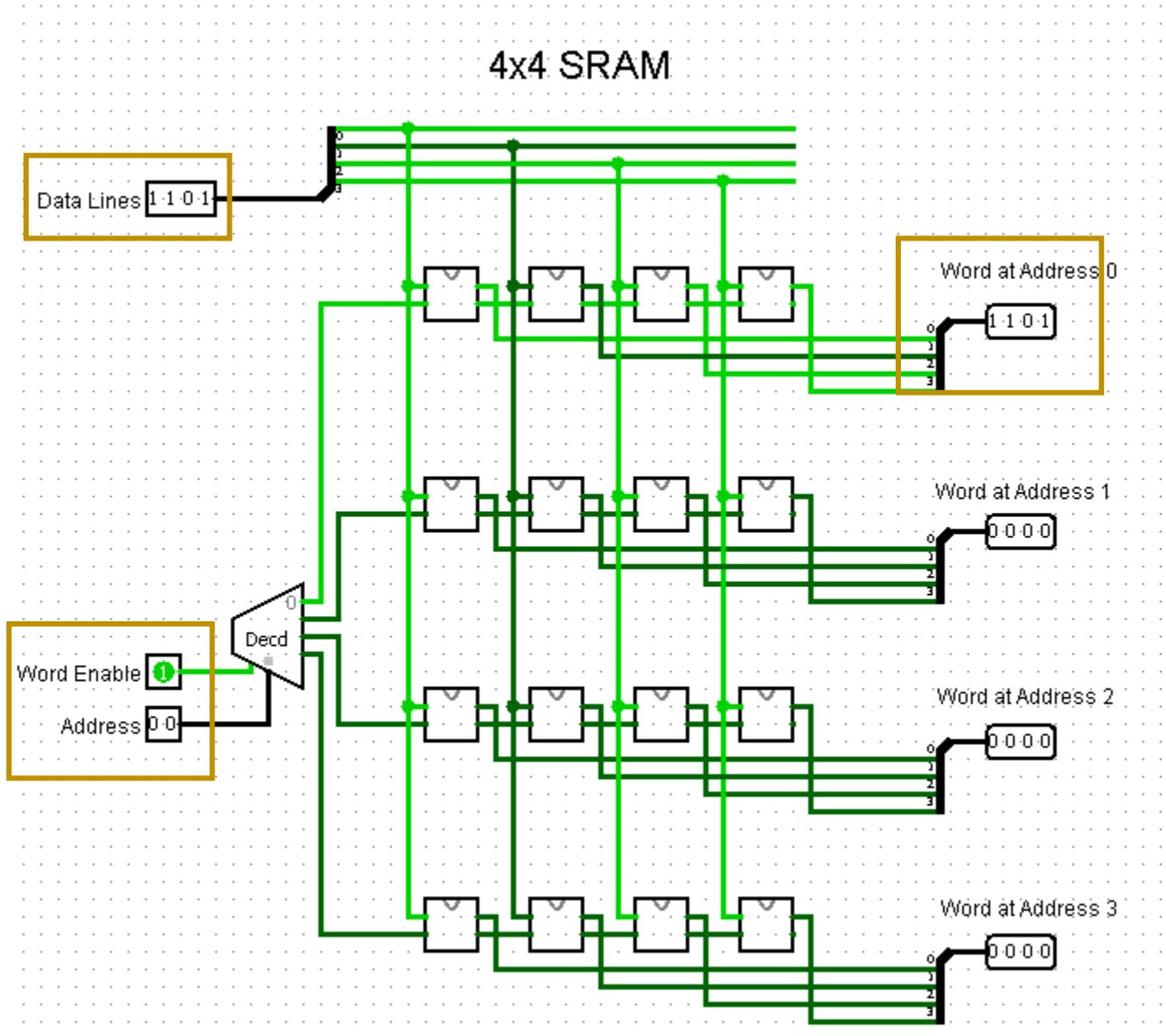

0
Decd
Word Enable
1
Address
0 0


0
Decd
Word Enable
1
Address
1 0

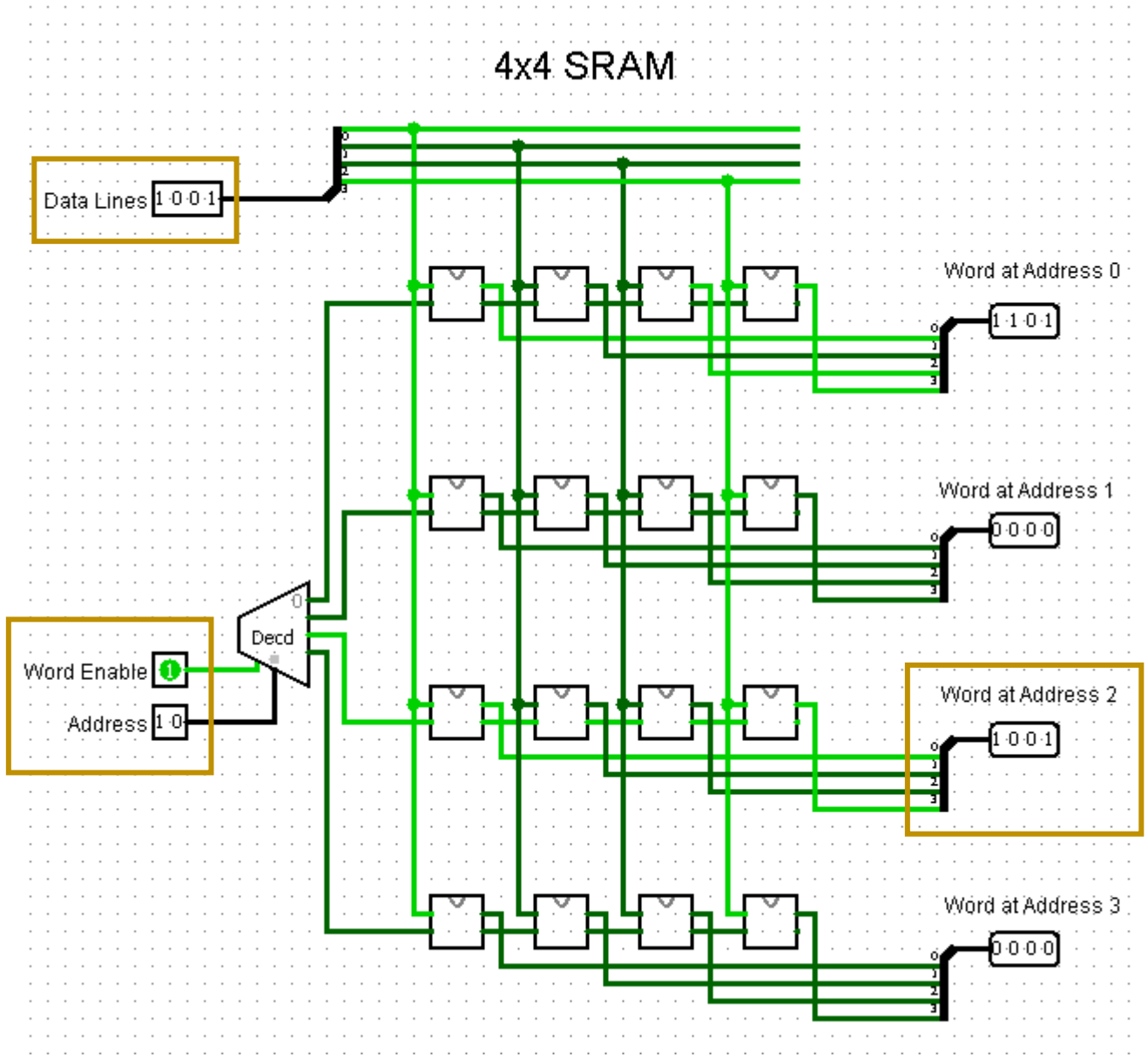

0
Decd
Word Enable
1
Address
0 1


0
Decd
Word Enable
1
Address
1 1

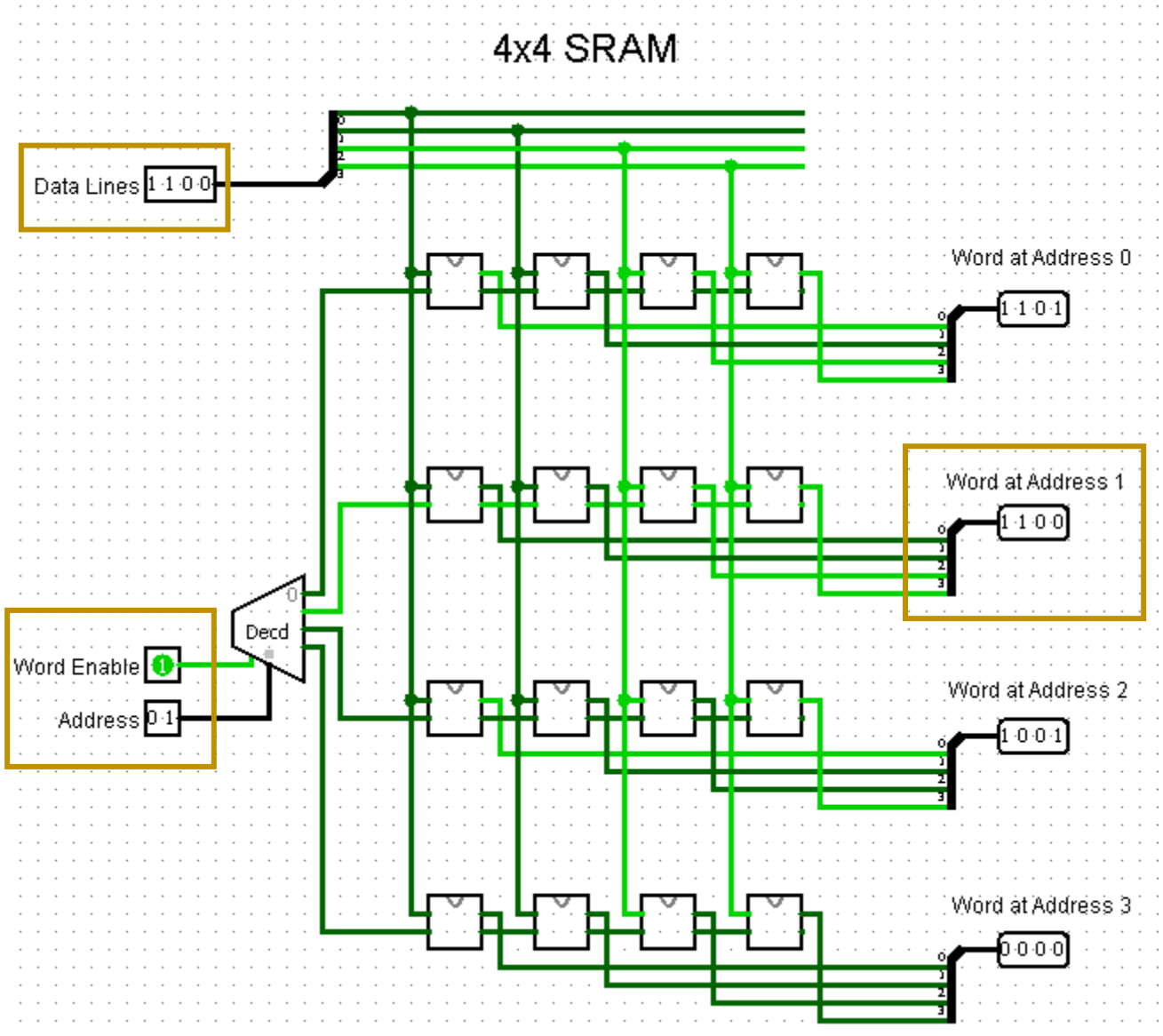
RAM


4x4 SRAM
Data Lines
1 1 0 1
Word at Address 0
1 1 0 1
Word at Address 1
0 0 0 0
Word at Address 2
0 0 0 0
Word at Address 3
0 0 0 0
Decd
Word Enable
1
Address
0 0

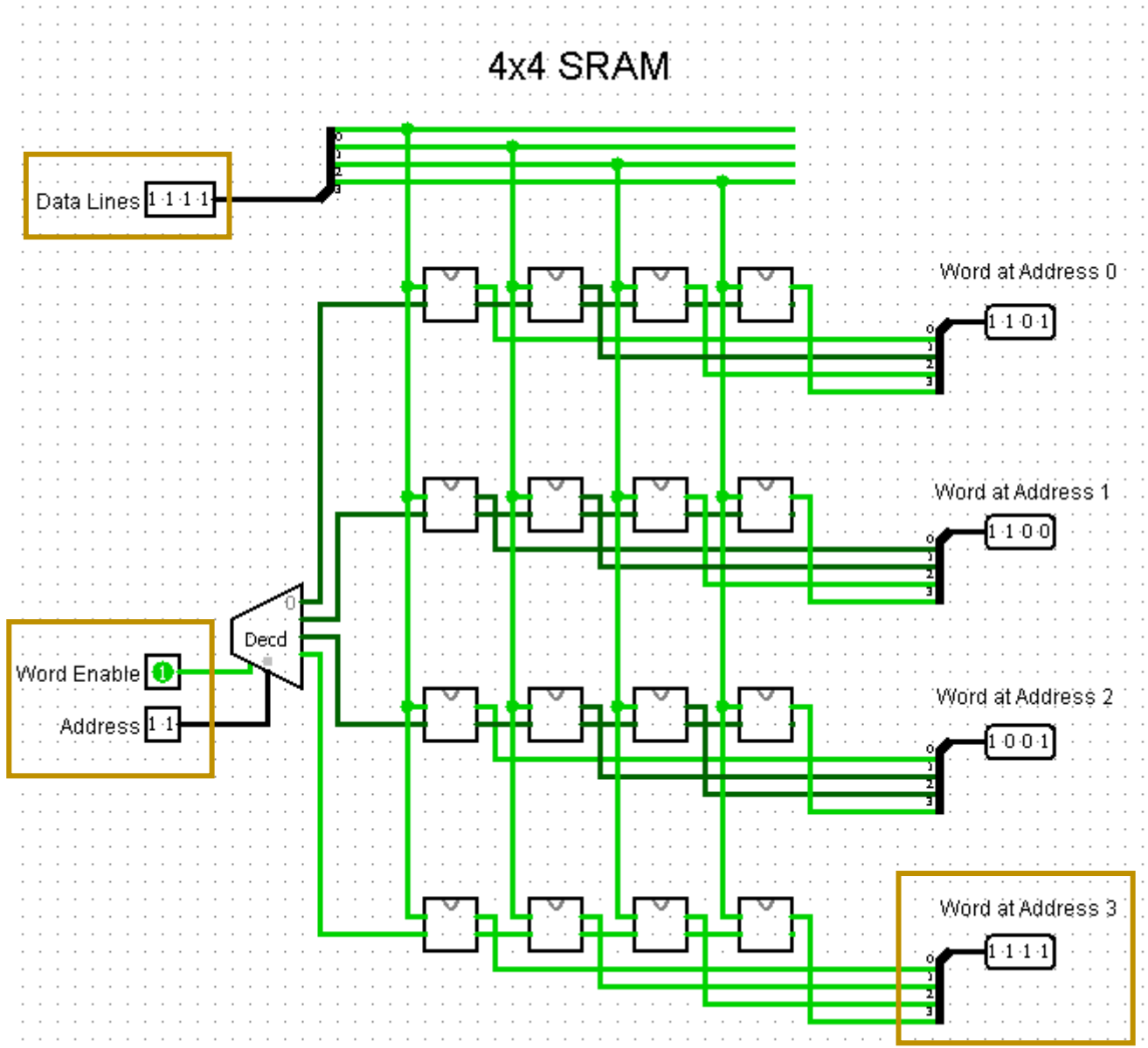
RAM


4x4 SRAM
Data Lines
1 0 0 1
Word at Address 0
1 1 0 1
Word at Address 1
0 0 0 0
Word at Address 2
1 0 0 1
Word at Address 3
0 0 0 0
Decd
Word Enable
1
Address
1 0

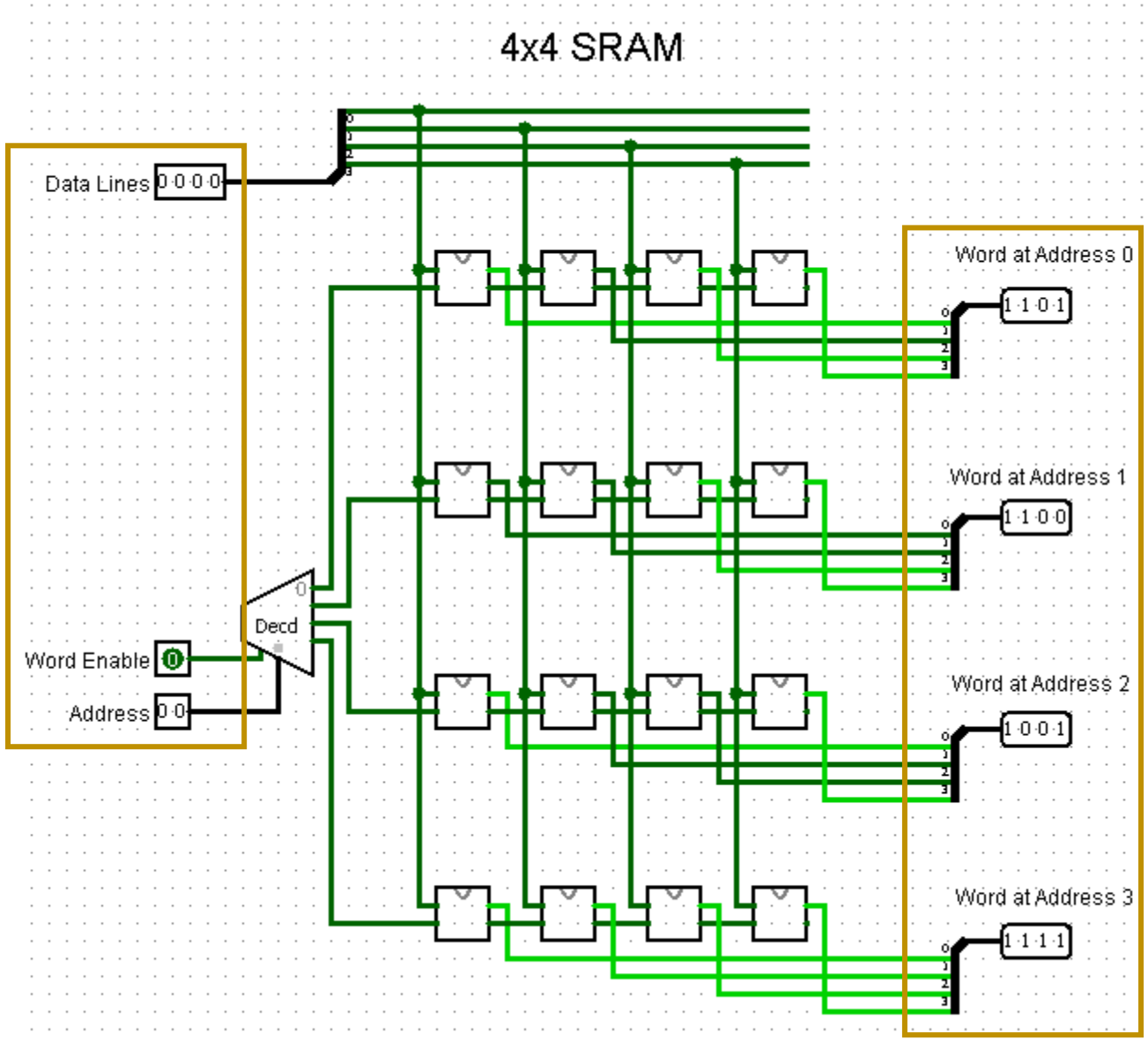
RAM


4x4 SRAM
Data Lines
1 1 0 0
Word at Address 0
1 1 0 1
Word at Address 1
1 1 0 0
Word at Address 2
1 0 0 1
Word at Address 3
0 0 0 0
Decd
Word Enable
1
Address
0 1

RAM


4x4 SRAM
Data Lines
1 1 1 1
Word at Address 0
1 1 0 1
Word at Address 1
1 1 0 0
Word at Address 2
1 0 0 1
Word at Address 3
1 1 1 1
Decd
Word Enable
1
Address
1 1

RAM


4x4 SRAM
Data Lines
0 0 0 0
Word Enable
0
Address
0 0
Decd
Word at Address 0
1 1 0 1
Word at Address 1
1 1 0 0
Word at Address 2
1 0 0 1
Word at Address 3
1 1 1 1

RAM

- Repeated reads and writes are required to refresh the capacitor in a DRAM cell
  - As opposed to the static storage of SRAM
  - This gives SRAM faster access than DRAM

- DRAM cells have fewer components and thus much smaller than SRAM cells
  - DRAM cells can be packed more densely than SRAM cells
  - DRAM is cheaper per bit than SRAM

RAM

- An improvement of DRAM is **S**ynchronous **DRAM** (**SDRAM**)
  - The clock keeps the processor and main memory synchronized.

- The use of a clock gives SDRAM the benefit of implementing pipelining
  - Pipelining allows a device to perform multiple operations at once
  - For example, SDRAM might output data to the processor while receiving the next address- simultaneously
  - We'll discuss pipelining in a later lecture

RAM

- An improvement of SDRAM allows two words to be written or read during a single clock cycle
  - One word when the clock is 1, and another word when the clock is 0

- This is referred to as **D**ouble **D**ata **R**ate (**DDR**) **SDRAM**

RAM

- **DDR SDRAM**
  - DDR2 doubled the data rate by allowing four words to be read/written during a single cycle
  - DDR3 quadrupled the data rate by allowing eight words to be read/written during a single cycle
  - DDR4 used a more advanced architecture for a higher transfer rate with a faster clock
    - Does not increase the words read/written during a single cycle
  - DDR5 (current technology)

ROM

- **ROM** (**R**ead-**O**nly **M**emory) is a type of non-volatile memory that retains its stored bits.

- ROM is used by technologies that write data slowly, meaning writing data to the ROM is less frequent than reading data from it.
  - Contrary to what its name suggests, writing to ROM (*"programming the ROM"*) is possible.

- ROM commonly uses a floating-gate transistor (FGT), where electrons remain trapped even when no longer powered
  - A large positive voltage traps the electrons
  - A large negative voltage releases the electrons

ROM

- There are several different types of ROM
  - **Masked-Programmed ROM**: The word line to bit line connections are hardwired and can never be changed
  - **O**ne-**T**ime **P**rogrammable **ROM** (**OTP ROM**): The word line to bit line connections have a fuse that, when blown, can break the connection; Can only be programmed once.
  - **E**rasable **P**rogrammable **ROM** (**EPROM**): Electrons are trapped in FGTs using a large positive voltage; Electrons are released when the chip is exposed to ultraviolet light

ROM

- **E**lectrically **E**rasable **P**rogrammable **ROM** (**EEPROM**): Electrons are trapped in FGTs using a large positive voltage; Electrons are released from FGTs using a large negative voltage.

- **Flash**: A type of EEPROM; Electrons are trapped in FGTs using a large positive voltage; Using a large negative voltage, electrons in entire blocks of FGTs are released at once.

EPROM