Memory System

- Provides storage for instructions and data
- Large storage capacity eases the task of developing programs
- Greater speed improves performance and reduces the need to stall the CPU
- Reduced cost makes the overall system more economical
- All of these goals cannot be achieved at the same time
- Techniques such as caching and virtual memory can give the illusion of greater speed and capacity

Memory System

- There are two basic types of memory
  - Read/write can be changed or updated (RAM)
  - Read-only can be read but not changed (ROM)

- Memory access is a read or write operation
  - Location to access must be specified
  - Read obtains copy of contents
  - Write replaces contents with specified data

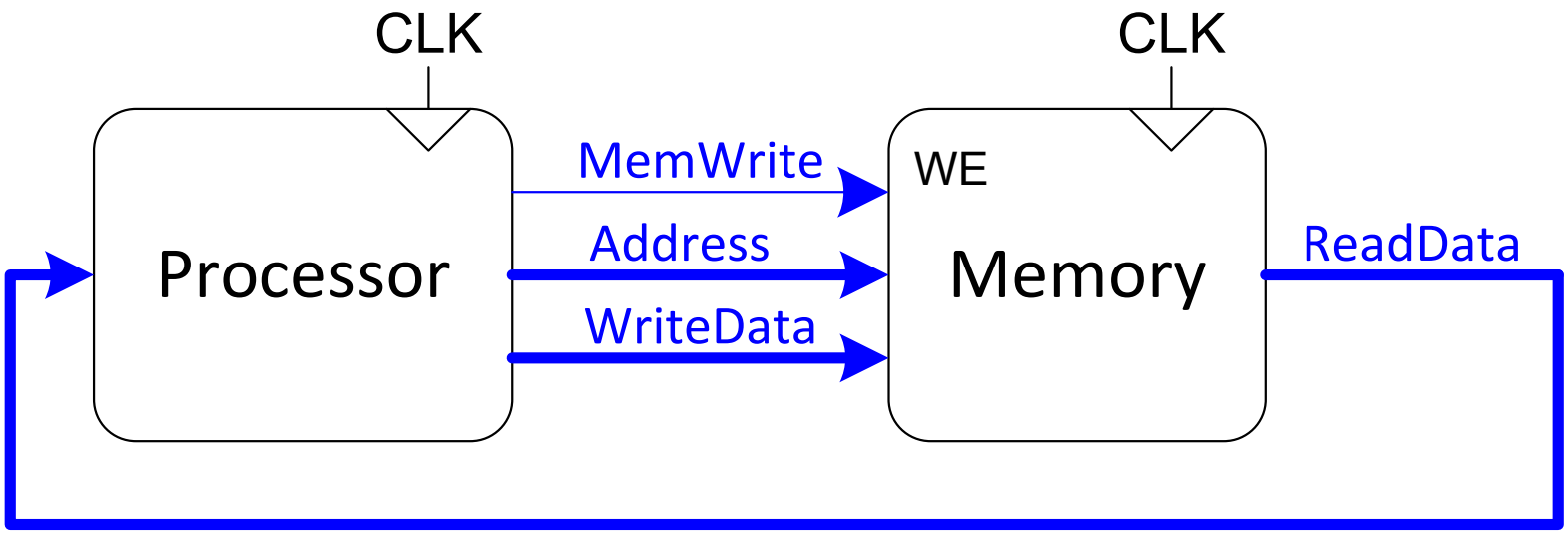
- Each location is assigned a unique number (address)

- Any location in RAM or ROM can be accessed directly

- Storage capacity is measured in units of 8-bit bytes

Memory Interface

Example: Processor writes to memory by specifying the address, the data, and the write control signal.



The clock signal (CLK) synchronizes the interactions.

Memory System

- Most systems are “byte addressable”
  - Individual bytes can be accessed
  - Actual transfer size matches bus width

- Multi-byte items usually must reside on proper boundary
  - Word (4 bytes) address must be multiple of 4
  - Half word (2 bytes) address must be even
  - Address of aligned data item is a multiple of its size
  - Unaligned items may require multiple transfers
  - Unaligned accesses cause exceptions on MIPS

Memory System

- MIPS memory features
  - Employs 32-bit addresses (4 GB address space)
  - Byte-addressable
  - Enforces memory alignment
- Amount of physical memory dictates number of address bits
- Width of pathway (bus) dictates number of bytes in a transfer
- Usually, at most 1 read or write can occur at a time

Memory System

- "Access time"
  - time between read request and return of data
- "Memory cycle time"
  - minimum time between consecutive reads
  - Includes setup time, access time and recovery time
- Addresses are sent over the address bus
- Data bits are sent over the data bus
- Read/write request signals are sent over the control bus

Memory System

- There are two types of RAM
  - Dynamic RAM (DRAM)
  - Static RAM (SRAM)

- DRAM stores charge to represent 0 or 1
  - Charges leaks off overtime
  - Requires periodic refresh to restore charge
  - Must be charged before a read occurs
  - Reads are destructive (must rewrite to restore)
  - Relatively inexpensive
  - Allows more bits per unit area (more dense)
  - “Volatile” contents lost when power is off

Memory System

- SRAM uses switches (gates) to store bits
  - Provides much shorter access time than DRAM
  - Contents remain stable as long as power is on
  - Reads are non-destructive
  - More expensive than DRAM
  - Consumes more area per bit than DRAM
  - Used for high speed memory (cache)
  - Volatile, contents lost when power is off

Memory System

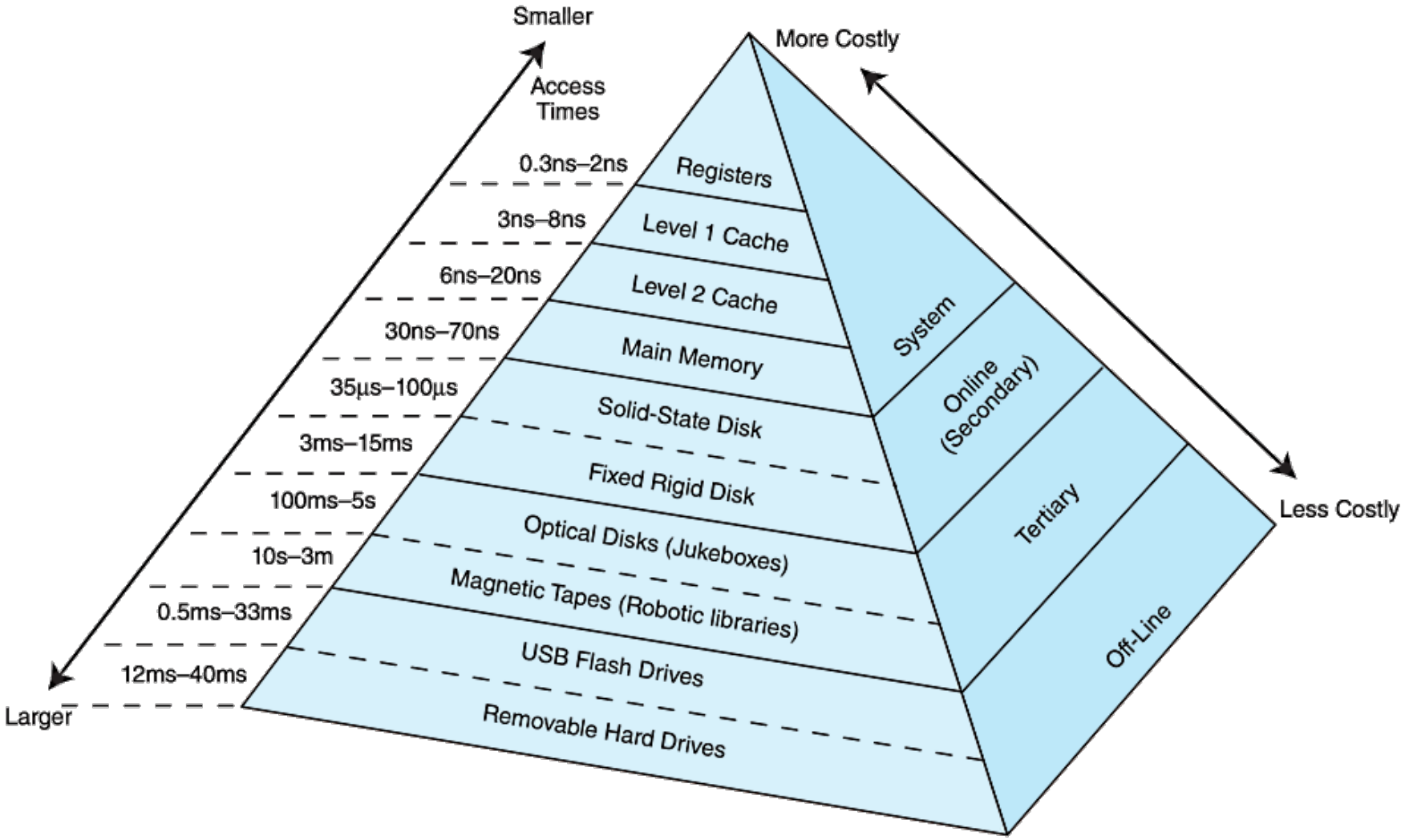
- ROM needs no refresh
  - Used to store permanent of semi-permanent data
  - Contents remains intact even when power is off
  - Reads are non-destructive
  - “Non-volatile” contents persists when power is off

- Other memory types
  - PROM (programmable) may be written once
  - EPROM (erasable PROM) exposed to UV to erase
  - EEPROM (electrically erasable) erase/rewrite inplace
  - FLASH (entire blocks must be erased)

Memory Hierarchy

- In general, faster memory is more expensive than slower memory
- Different types of memory can be arranged in a hierarchy
- Small fast storage elements (registers) are kept in the CPU
- Cache is slightly slower than registers and kept close to the CPU
- Larger slower main memory is accessed through the bus
- Even larger and much slower storage (disks, tapes, network drives) are farther from the CPU

Memory Hierarchy



Memory Addressing

- Most systems are byte addressable
  - Each 8-bit byte is assigned a unique number (address)
  - Addresses range from 0 to some maximum
  - Consecutive byte addresses differ by 1
- Maximum address depends on address register width
- Larger storage units consist of multiple bytes
  - Words (4 bytes), half words (2 bytes)
  - Word size matches the CPU register size

Memory Addressing

- Some systems may be word addressable
  - Each word contains multiple bytes
  - Consecutive word addresses differ by 1

- MIPS processor uses 32-bit registers and addresses
  - Addresses range from 0 to $2^{32} - 1$
  - Registers and words contain 4 bytes

Memory Byte Order

- SW instruction copies a register into memory
  - Leftmost byte is stored at the lowest address
  - Next lower byte is stored at the next higher address
  - This “big endian” storage order is used by the MIPS

- Example: if 0x12345678 is stored at address 200:

Address	200	201	202	203
contents	0x12	0x34	0x45	0x78

Memory Byte Order

- Other systems use “little endian” memory storage
  - Rightmost byte is stored at the lowest address
  - Next higher byte is stored at next higher address
  - Intel systems use this memory storage order

- Example: if 0x12345678 is stored at address 200:

Address	200	201	202	203
contents	0x78	0x56	0x34	0x12

Memory Byte Order

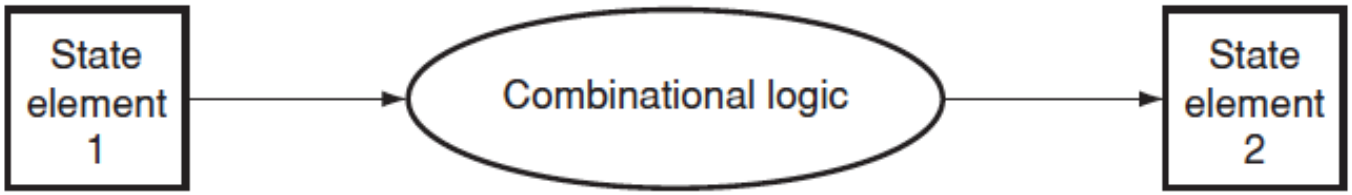
- Byte order matters when exchanging data
  - Network order is big endian
  - Little endian systems must reorder network bytes received
- Registers always contain bytes in high to low order
  - High byte on left, low byte on right
- Character strings are arrays of bytes
  - Individual bytes are accessed
  - Characters in string are ordered from first to last
  - Address of string = address of leading character
- Byte order matters when accessing multi-byte items

Conclusion

- We will examine registers, cache, main memory, and virtual memory.
- Registers are accessed directly by the processor
  - Instructions contain register numbers (rs, rt and rd)
  - Registers are contained within the CPU
- Virtual memory extends the address space from RAM (main memory) to the secondary storage (hard drive)
- Virtual memory provides more space: Cache memory provides speed

Memory Implementation

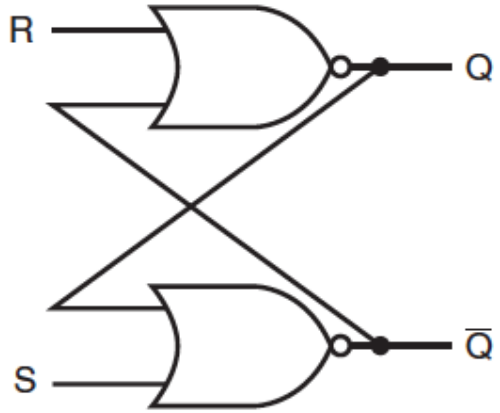
- Combinational logic circuits need state elements to:
  - provide inputs
  - store the output
- State elements are sequential logic devices
  - Their output depends on the history of the inputs
- Combinational logic output depends only on current inputs



- State changes occur at clock edges

S-R Latch

A 1-bit memory device must capture and store its input



Cross coupled NOR gates

S and R are the inputs

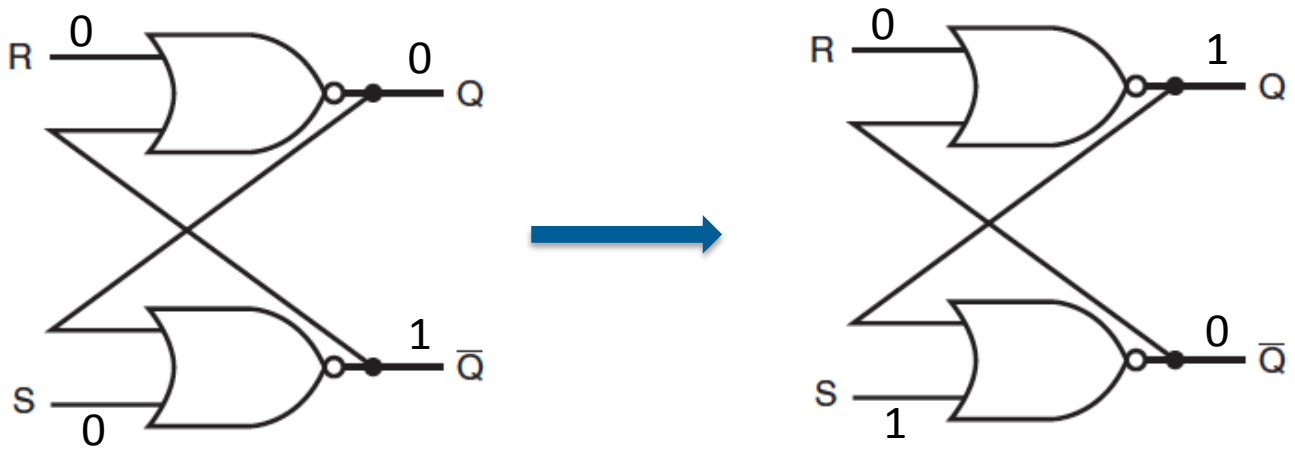
The value of the output Q defines the state (0 or 1)

$\overline{Q}$ (NOT Q) is the other output

Outputs do not change as long as S and R are both 0

S-R Latch

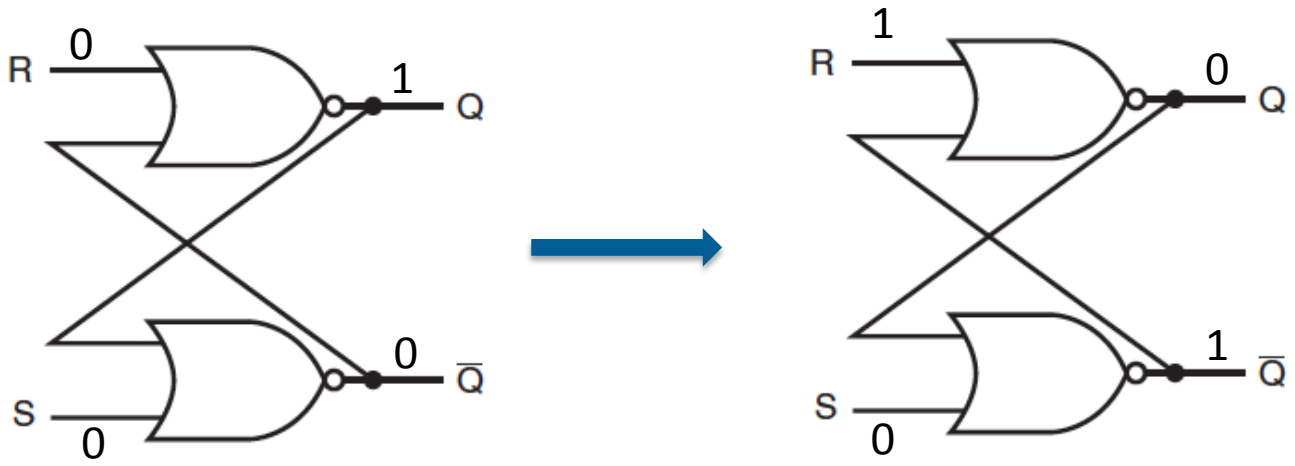
In state 0, if S changes from 0 to 1, Q becomes 1



When S goes back to 0, Q retains its new value of 1

S-R Latch

In state 1, if R changes from 0 to 1, Q resets to 0



When R goes back to 0, Q retains its new value of 0

S-R Latch

Behavior of the S-R Latch is described by a *characteristic table*
Defines output at t+1 (next cycle) as function of inputs at t

S	R	Q(t+1)
0	0	Q(t) no change
0	1	0
1	0	1
1	1	? unpredictable

Differs from truth table which shows output for current cycle

S-R Flip-Flop

S-R Latch

- Sets Q = 1 if S is asserted while clock is asserted
- Sets Q = 0 if R is asserted while clock is asserted

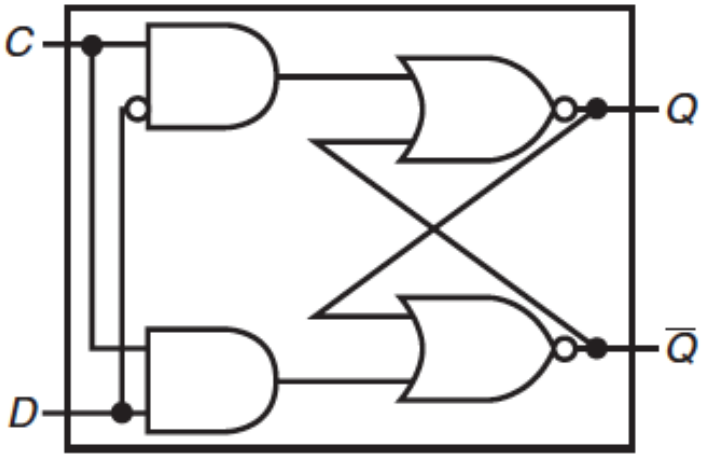
S-R flip-flop

- Sets Q = 1 if S is asserted at clock edge
- Sets Q = 0 if R is asserted at clock edge

D Latch

What is needed for single bit storage device is a D latch or flip-flop

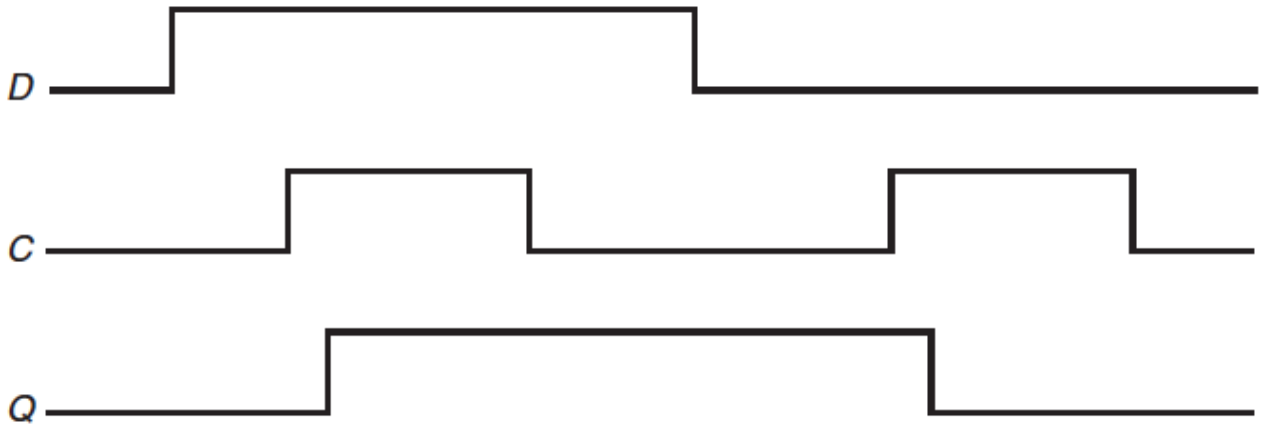
D Latch has single data input and a clock input



The two AND gates open when C (clock) is high

Setting D=1 has same effect as S=1 and R=0 with S-R latch
Setting D=0 has same effect as R=1 and S=0 with S-R latch

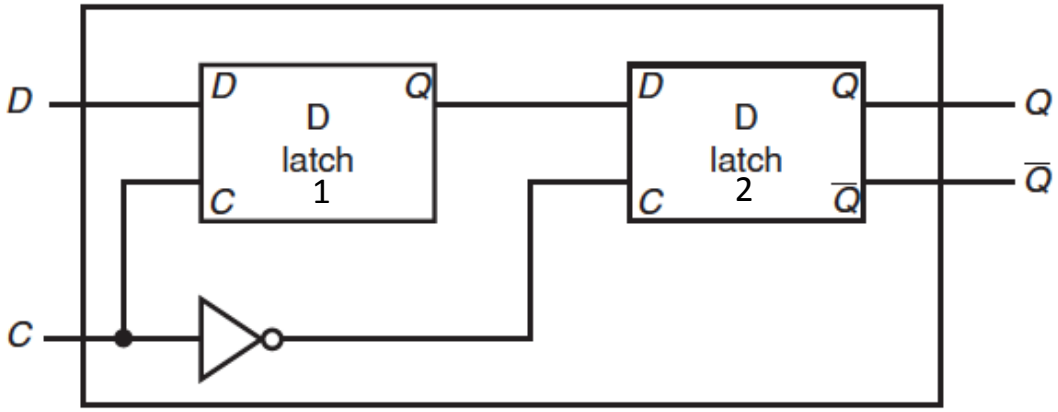
D Latch Behavior



When C, the clock, is high, Q takes on the same value as D

D Flip-Flop

D flip-flop (with falling edge trigger)



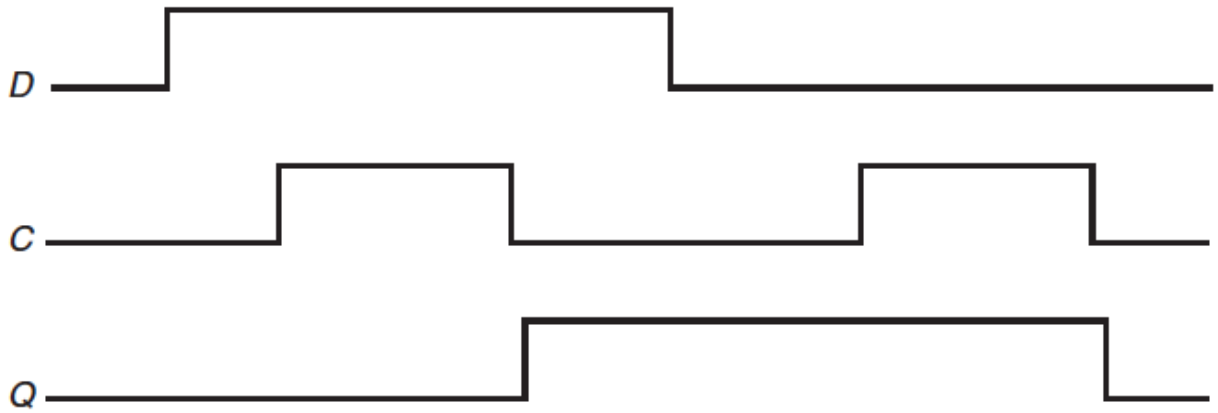
When C is high, latch 1 passes its input (D) to its output

When C goes low:

- output from latch 1 is retained
- Latch 2 passes output from latch 1 on to latch 2's output

Q only changes when clock goes high and back low

D Flip-Flop Behavior



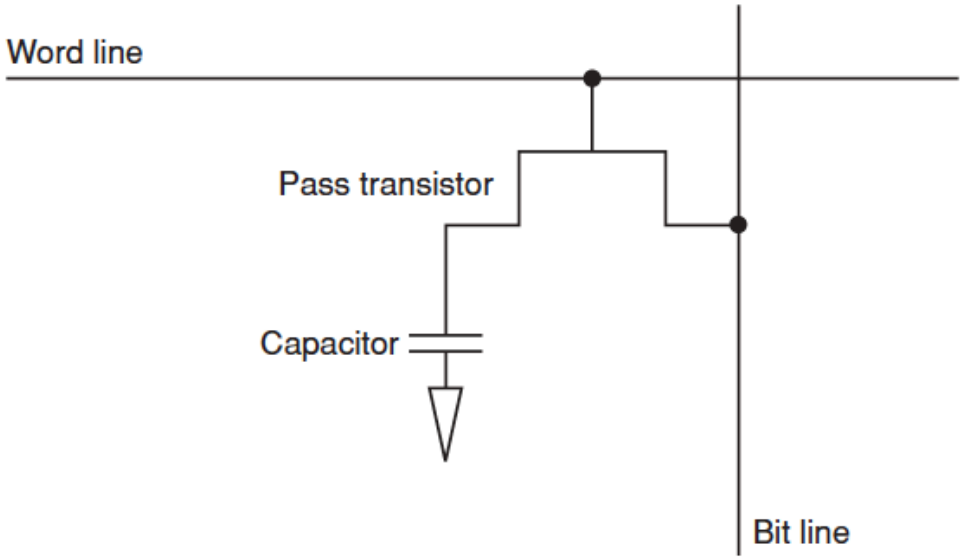
Output Q changes at trailing clock edge in this example

The alternative would be leading edge trigger

As long as power is applied, the stored bit is retained

DRAM

With dynamic RAM, the bit is stored as charge on a capacitor
A transistor switch allows the bit to be read or written



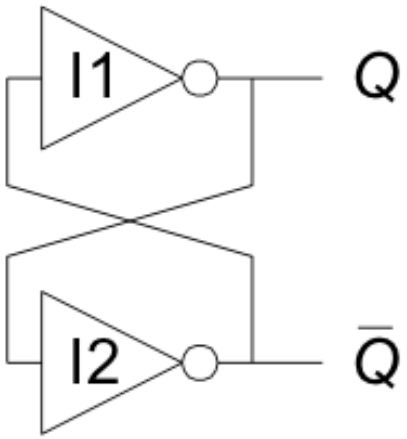
Asserting word line closes switch allowing access via bit line

Presence of charge represents 1
Absence of charge represents 0
Must refresh every few milliseconds to maintain the charge

1- Bit Storage Cell

Storage cells must have 2 stable states: 0 and 1 (*bi-stable*)

Another possible implementation uses cross-strapped inverters:

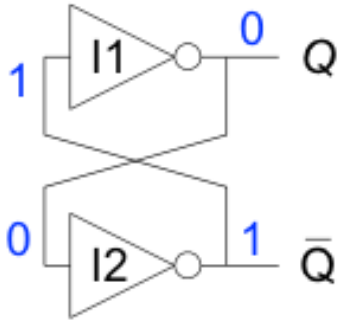


Stable output Q defines the state
State does not change

1- Bit SRAM

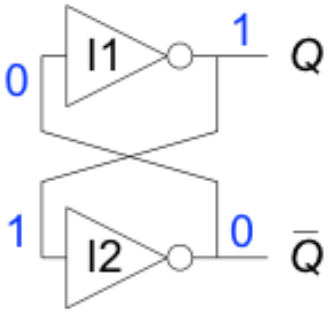
$Q = 0$:

then $\overline{Q} = 1$, $Q = 0$ (consistent)



$Q = 1$:

then $\overline{Q} = 0$, $Q = 1$ (consistent)

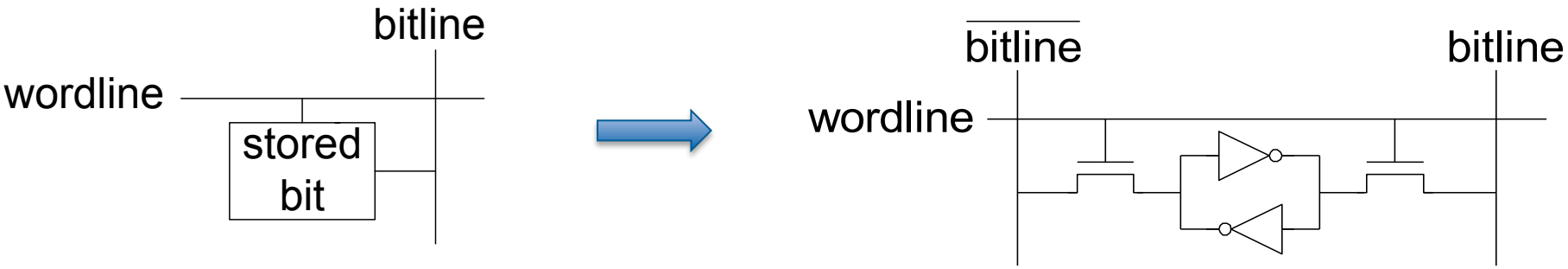


Stores 1 bit of state in the state variable, Q (or $\overline{Q}$)

No refresh is required

1- Bit SRAM

Wordline and bitline allow access to the stored bit



Both switches turn on when wordline is asserted
Stored bit can be transferred to or from bitline or $\overline{\text{bitline}}$

Memory Comparison

Larger storage cells can be built using these single-bit devices

Flip-flops require more transistors to build

Cost, power and area consumed increases with more transistors

Memory Type	Transistors per bit	Latency
Flip-flop	20 or more	fast
SRAM	6	medium
DRAM	1	slow

Memory Comparison

Latency = time required to perform a read or write

Throughput = number of bits that can be accessed per unit time

Flip-flops have very short latencies and high throughput

SRAM has higher throughput and lower latency than DRAM

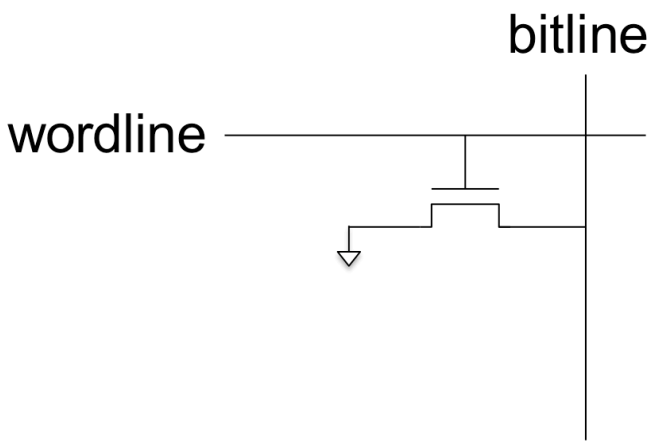
DRAM must wait for charge to move to bitline

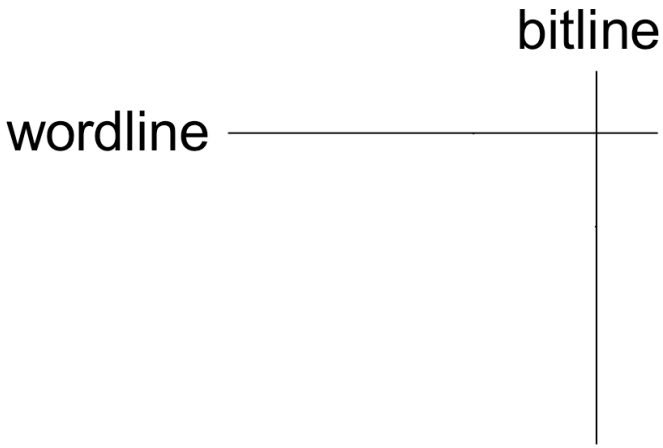
DRAM must be refreshed periodically and after each read

In general, latency increases with larger memory sizes

ROM

Bits are stored as the presence or absence of a transistor
Bitline is weakly pulled HIGH, then wordline is turned on





Transistor pulls bitline LOW if present (represents 0)
If transistor is absent, bitline remains HIGH (represents 1)
Nonvolatile, does not change if power is turned off

ROM Bit Storage Cell

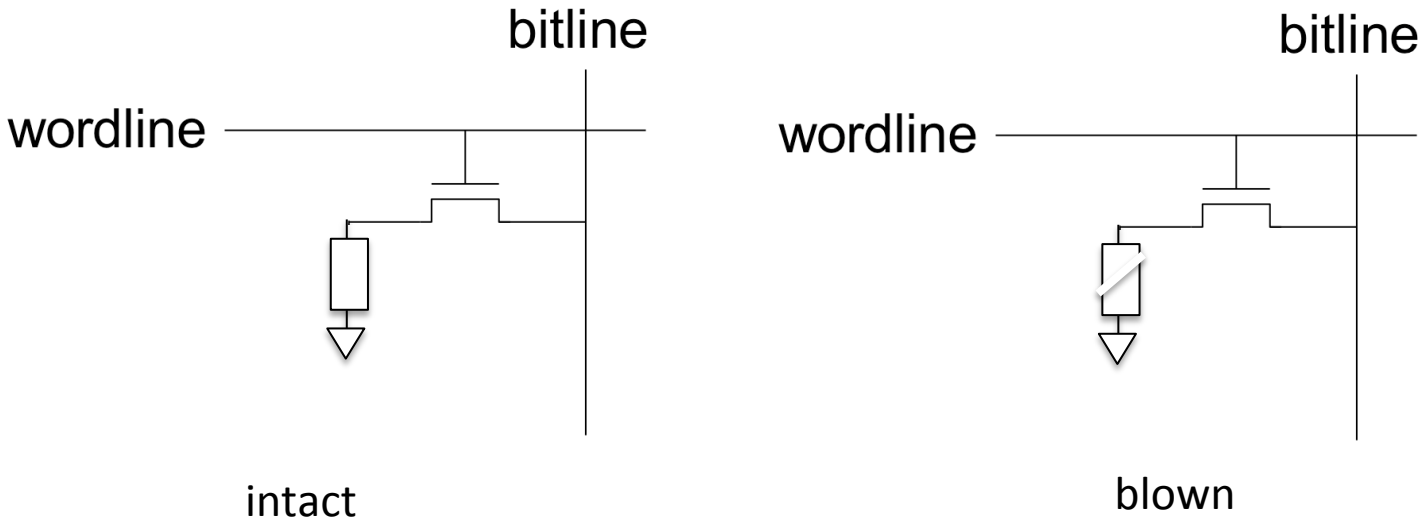
Contents of ROM bit cells can be set during manufacturing
- by including or omitting a transistor in various cells

These are sometimes called “masked” ROMs

PROMs (programmable ROM) have a transistor in every bit cell
- provides a way to connect or disconnect each transistor to ground

Fuse-PROM

High voltage is applied to selectively blow the fuse links
Bitline is pulled low if link is in place, otherwise it is high



Transistor pulls bitline LOW if fuse is present (represents 0)
blown fuse disconnects transistor from ground (bitline=1)
These are sometimes called "*one-time programmable*"

Erasable PROM

Some types or PROM are reprogrammable
- Transistors can be reversibly connected or disconnected to ground

Erasable PROMs (EPROM) use floating-gate transistors
- Electron tunnelling turns on the transistors when voltage is applied
- Exposure to UV light is used to erase the PROMs
- This requires removing the memory chip from its socket

Electrically erasable PROMs (EEPROMs) are erased in place
- EEPROM includes on-chip erasure circuitry

EEPROM bit cells are individually erasable

FLASH Memory

Flash memory is similar to EEPROM

Flash memory erases larger blocks rather than individual cells

Fewer erasing circuits are required for Flash

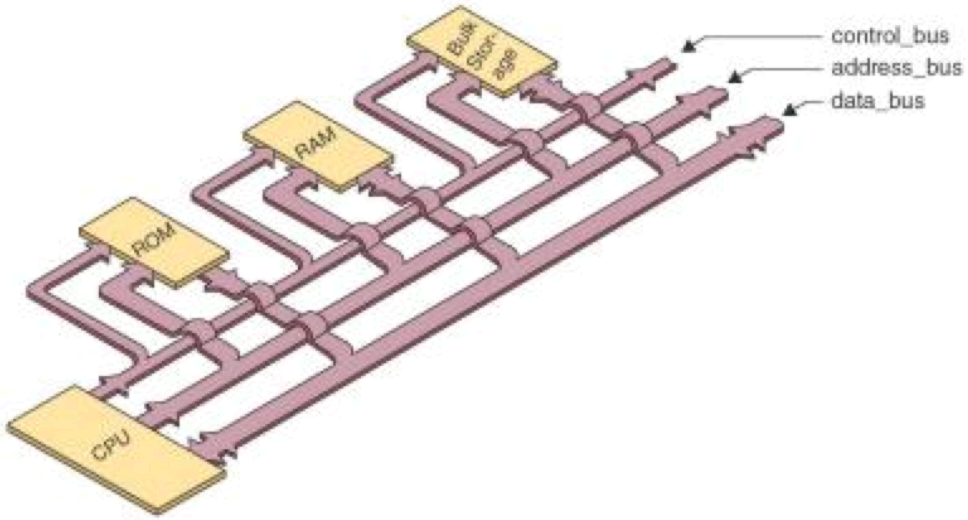
This makes Flash less expensive than EEPROM

Flash is a popular way of storing large amounts of data
Used in portable battery-powered cameras and music players

The various types of ROM take longer to write than does RAM

Memory Organization

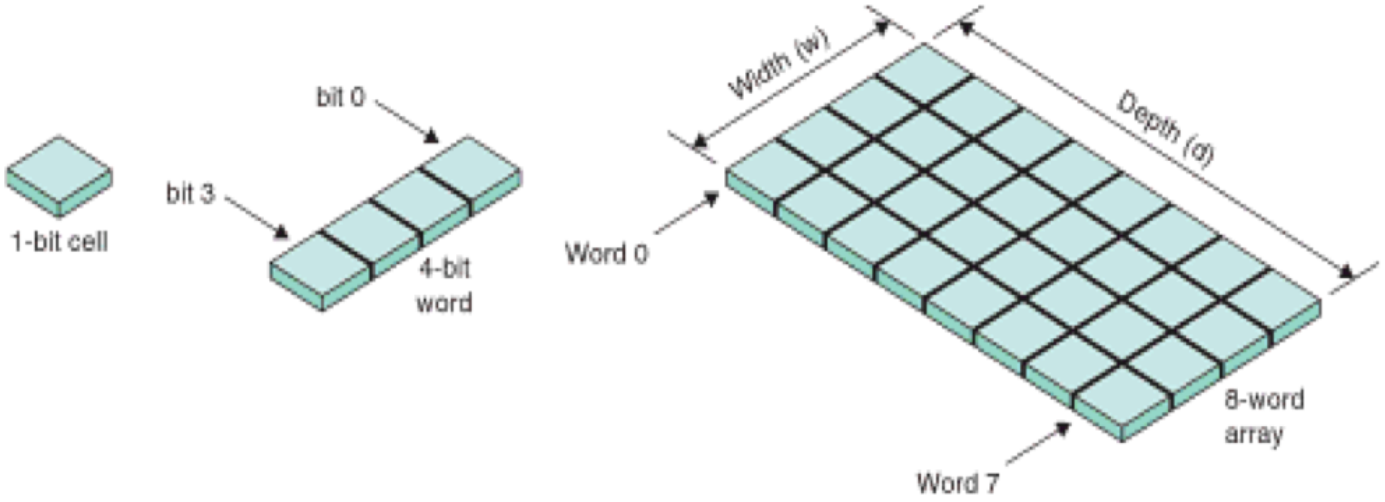
- The CPU obtains instructions and data from memory
  - Sends address over address bus
  - Sends or receives data over data bus
  - Sends read/write and other control signals over control bus



- A bus is a set of wires or electrical traces

Memory Organization

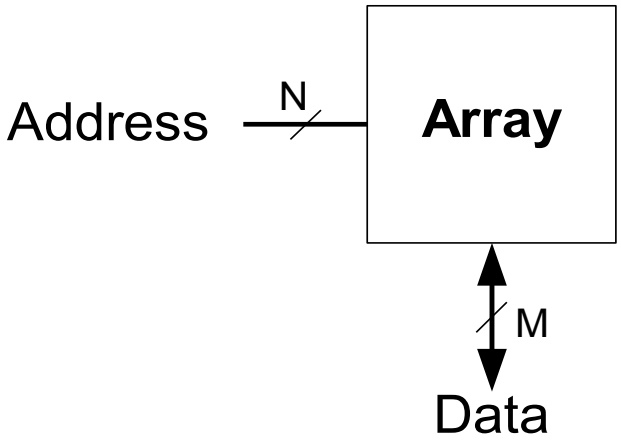
- A memory word is a group of 1-bit storage cells
- Each word has a unique address (from 0 up to some maximum)



- Number of bits per word defines its width
- Number of words in memory device defines its depth or height

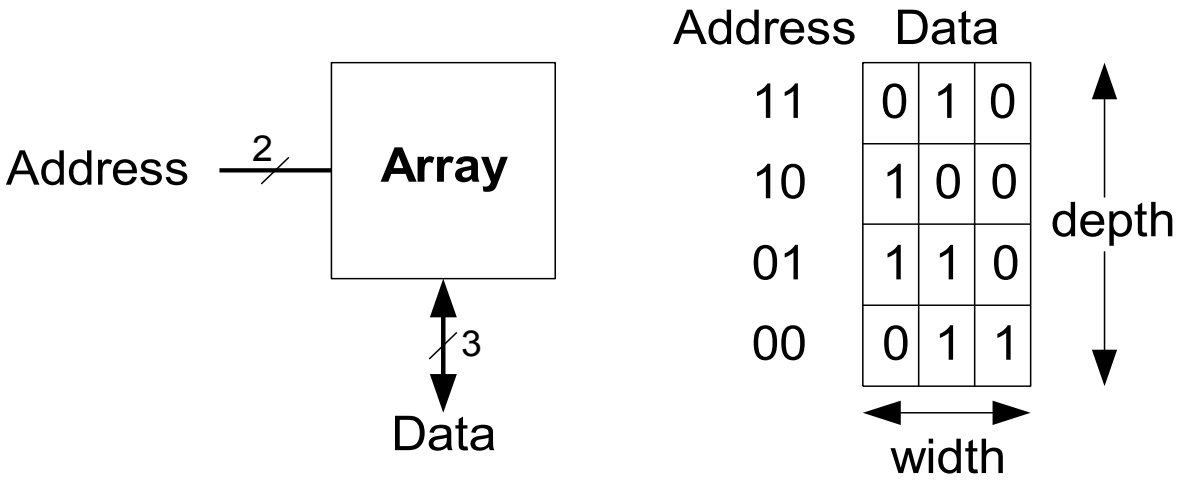
Memory Array

- The previous diagram shows a memory array
- The array contains 3-bit rows (words)
- In general, the array size = $2^N \times M$
  - N is the number of bits in the address
  - M is the number of bits per word



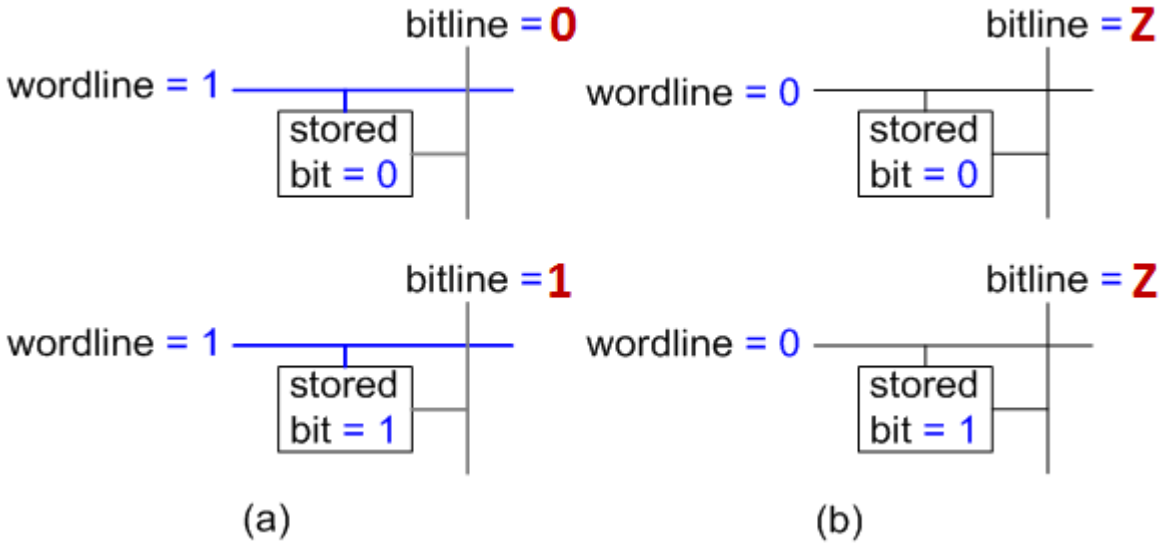
Memory Array Example

- $2^2 \times 3$-bit array
- Number of words: 4
- Word size: 3-bits
- For example, the 3-bit word stored at address 10 is 100

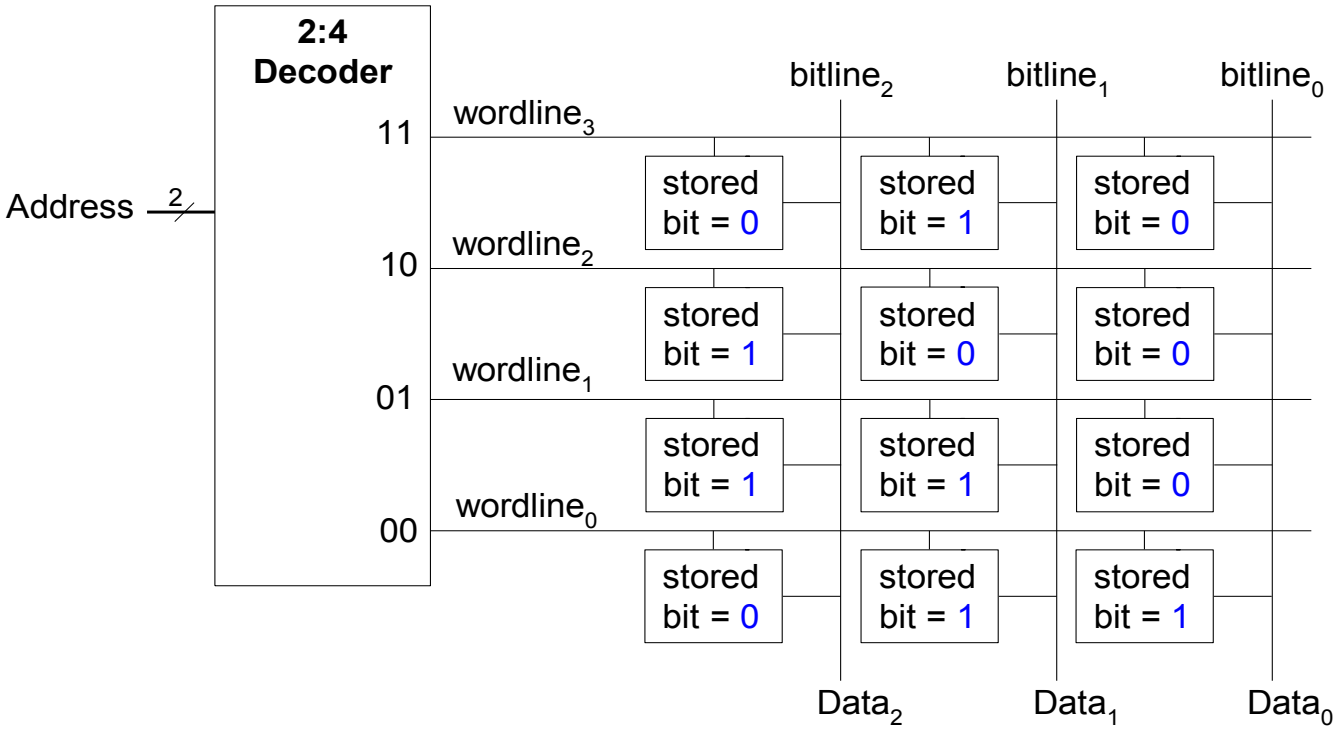


Memory Array Bit Cells

Wordline selects all bit cells within the specified word
Stored bit is transferred to or from bitline if wordline=1
Bits cells disconnected from bitline if wordline=0
Z represents the disconnected state

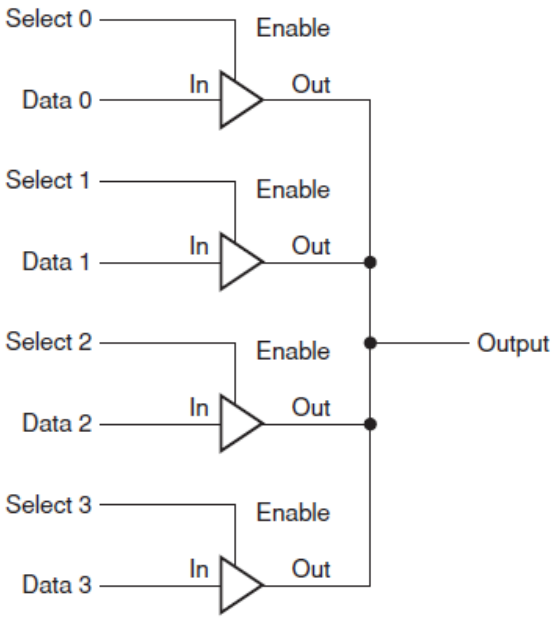


- **Wordline:**
  - like an enable
  - single row in memory array read/written
  - corresponds to unique address
  - only one wordline HIGH at once



Tri-State Buffers

The unselected words are detached from the bitlines

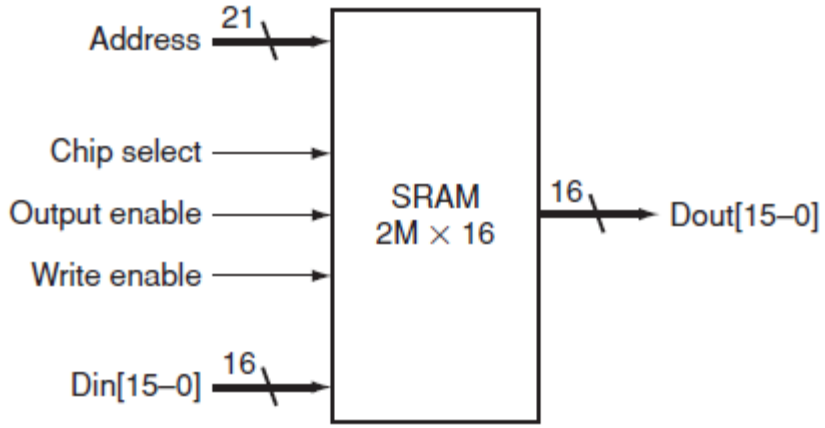


When enabled the tri-state buffer passes the data input through
When disabled, the tri-state buffer has a high impedence output
The output is thus disconnected from the bitline

Larger Memories

- Each memory array is contained in a separate chip
- Multiple chips are grouped to provide the desired memory size
- Chip select signals determine which chip to access
- The read/write enables determine the direction of data transfer
- The address determines which word or row to access
- Write data goes through data-in port
- Read data is copied out through data-out port
- Each chip performs only one read or write at a time

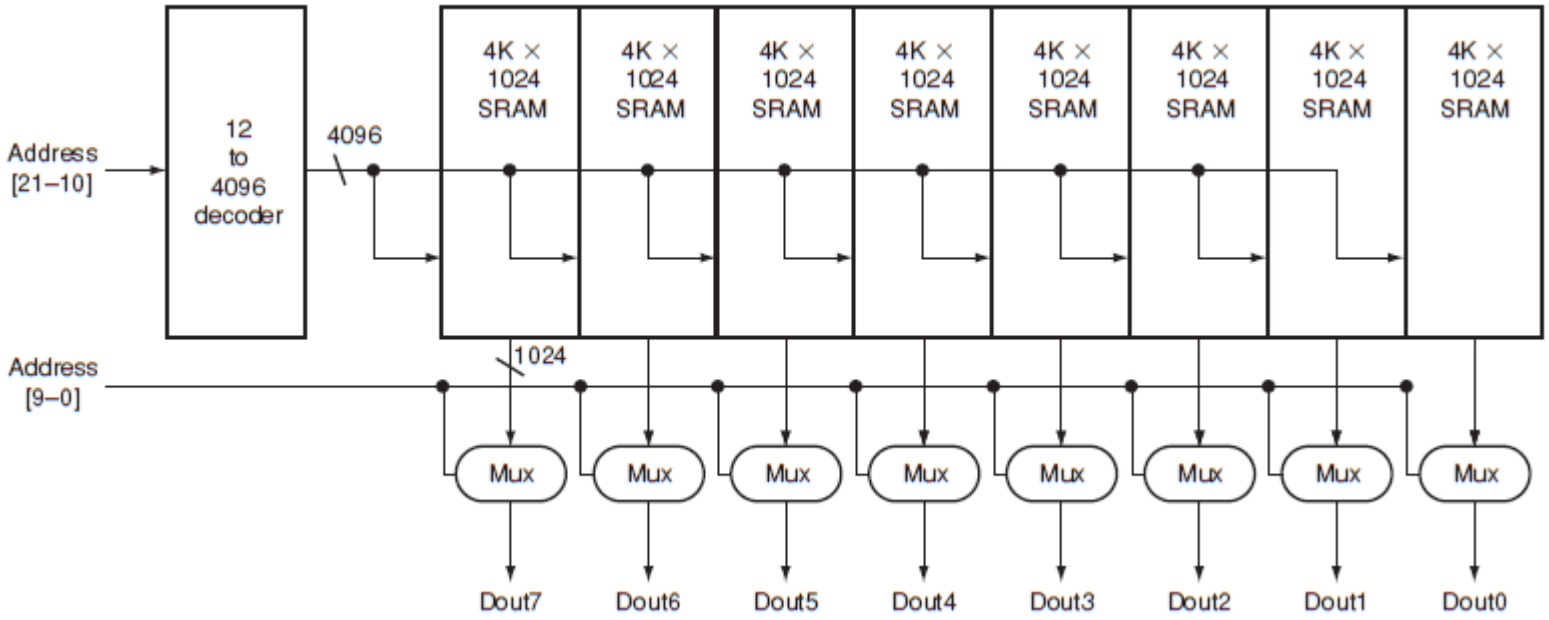
SRAM Example



- 2M (2^{21}) words, each 16 bits (2 bytes) wide
- Total size = 2097152*2 = 4194304 bytes
- A decoder could map the address onto the proper wordline

SRAM Example

- A 21-to-2097152 address decoder is needed in the previous case
- A more practical approach uses a 2-step decoding scheme



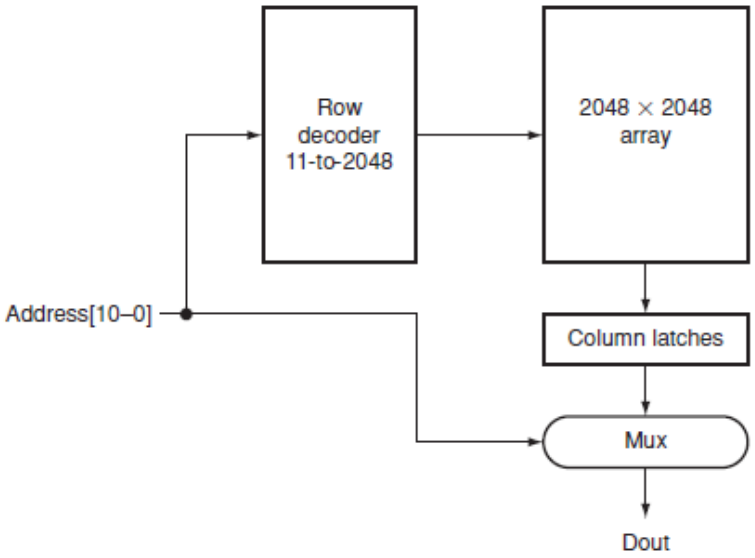
- 12-to-4096 decoder selects same row in every array
- 1024-to-1 Mux selects bit from each of the columns

DRAM Arrays

- DRAMs use a two-level decoding scheme to save on cost
- Same lines carry the row address and later the column address
- Row address determines which wordline is asserted
- Column address selects the data from the column latches
- Uses Row address (RAS) and Column address (CAS) strobes
- Refresh reads columns into latches and rewrites same values
- Entire row is refreshed in one cycle
- Memory controller handles refresh independently of CPU

DRAM Example

4M x 1 DRAM



11-bit row address selects one of 2048 rows (RAS)
Entire 2048-bit wide row is read into column latches
11-bit column address then selects one of 2048 latches (CAS)

Synchronous RAM

SDRAM and SSRAM transfer data in bursts

Burst is defined by starting address and a length

A clock is used to transfer successive bits in the burst

Multiple transfers occur without having to update the address

Significantly improves the rate of data transfer

SDRAM is the most popular choice for central memory

Other DRAM Types

DDRAM stands for double data rate RAMs (DDR)

DDRAM transfers data on both the rising and falling clock edge

DDR was standardize in 2000 and ran at 100 to 200 MHz

Later standards use increasingly higher speeds (≥ 1 GHz)

DDR2, DDR3 and DDR4

Memory Interleaving

A single memory module can perform 1 read or 1 write at a time

This limits performance, especially for narrow widths

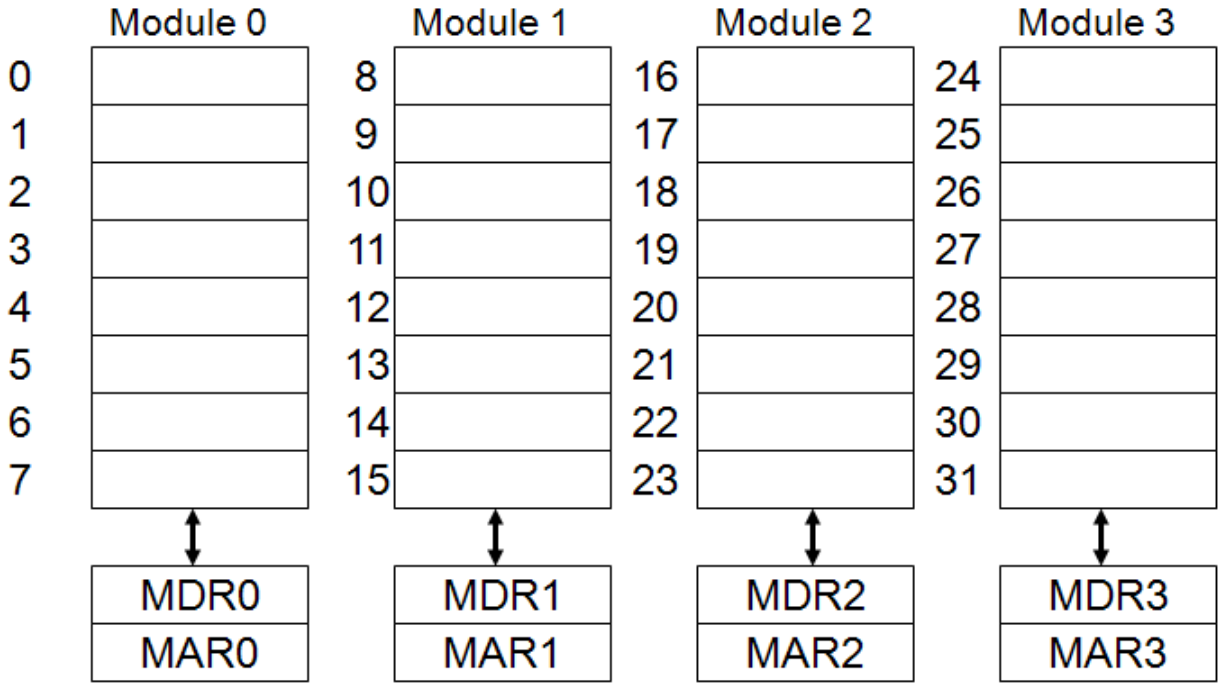
Interleaving refers to how consecutive locations are distributed

High order interleaving
- consecutive locations are in the same module
- the high order address bits indicate the module number

Consecutive locations must be accessed sequentially

Memory Interleaving

Example: if each module is 8 bits wide, reading a 4-byte word could require 4 separate reads from the same module.



MAR is memory address register.

MDR is memory data register.

Memory Interleaving

Low order interleaving

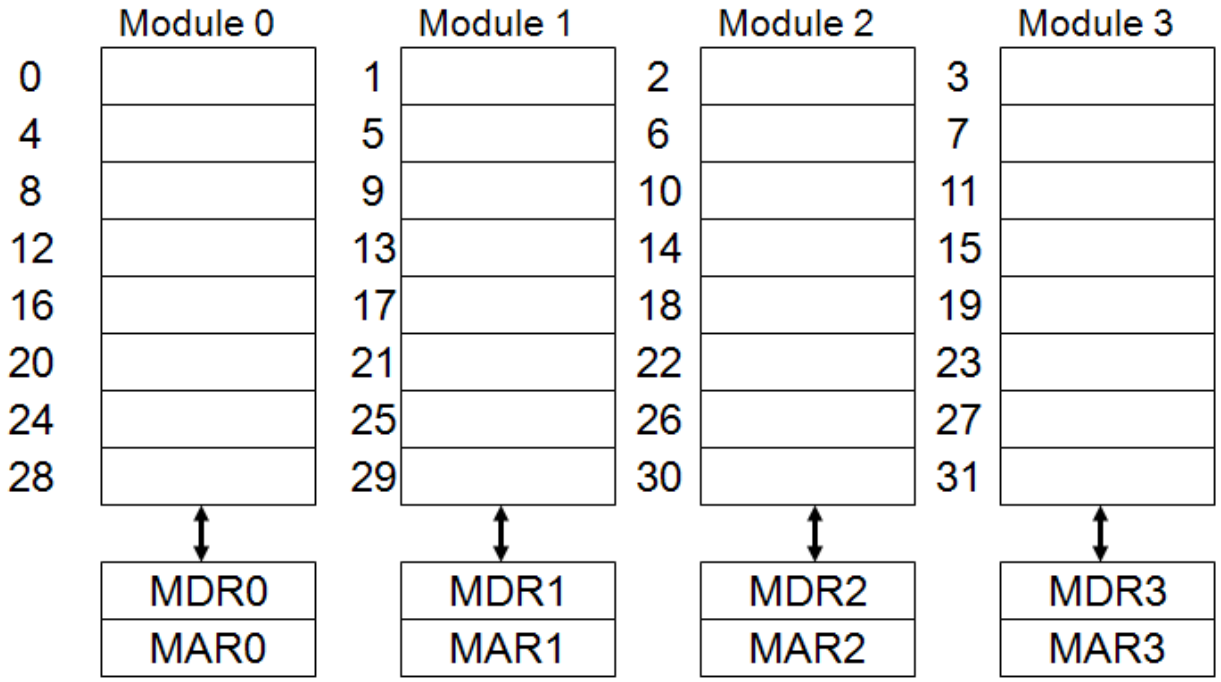
- consecutive locations are in different modules
- low order address bits indicate the module number
- high order address bits give offset into module

Consecutive locations can be accessed in parallel

High performance systems favor low order interleaving

Memory Interleaving

Low order interleaving example:



Bytes within each word have the same offset

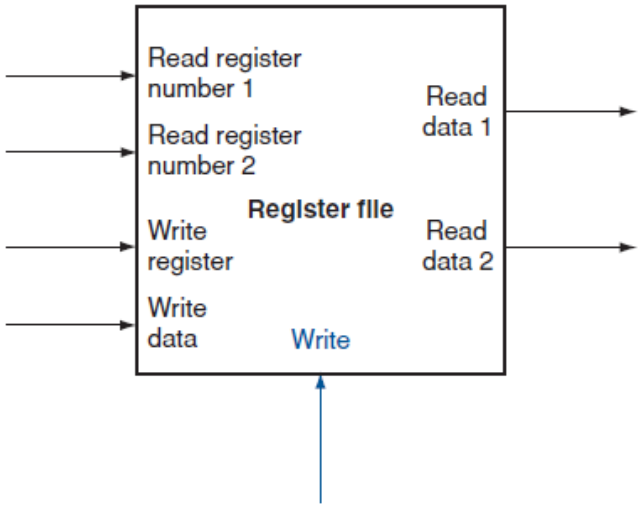
All four modules are read in parallel to obtain the word

The Register File

A set of registers each of which can be specified by a number

MIPS register file has two read ports and one write port

D flip-flops are used to construct the registers

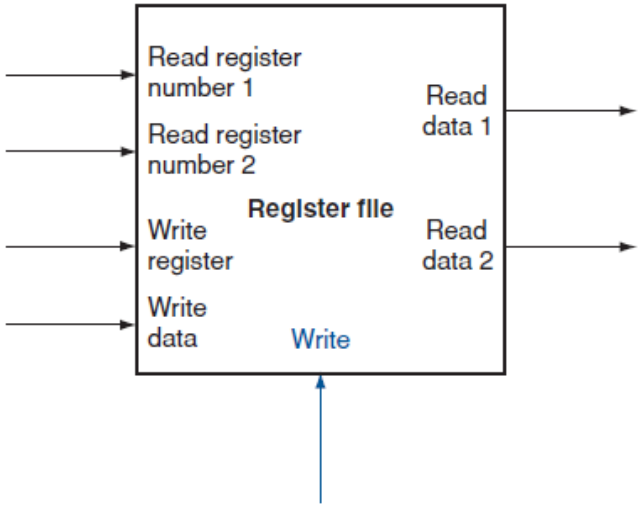


Write control line causes write data to be placed into write register

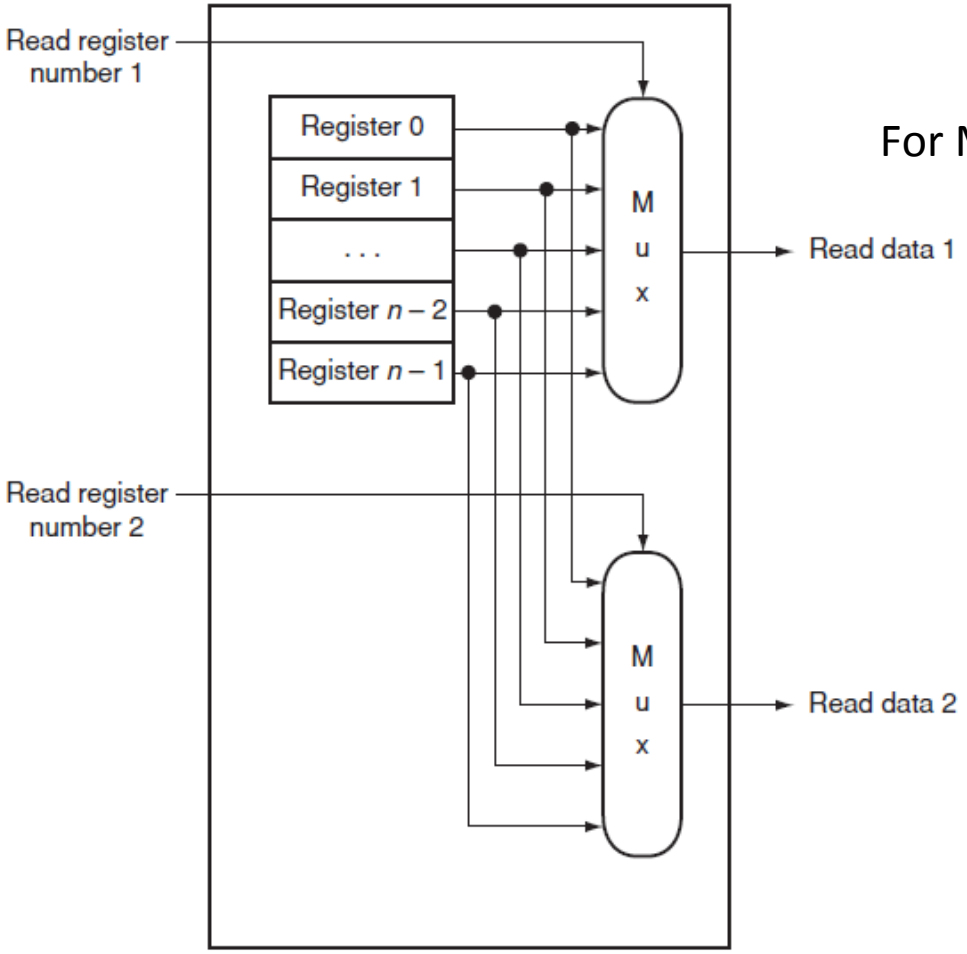
The Register File

5-bit read reg. number 1 identifies the rs register
5-bit read reg. number 2 identifies the rt register
5-bit write reg. identifies register to write (rd or rt)

Read data 1, read data 2 and write data are each 32-bit values



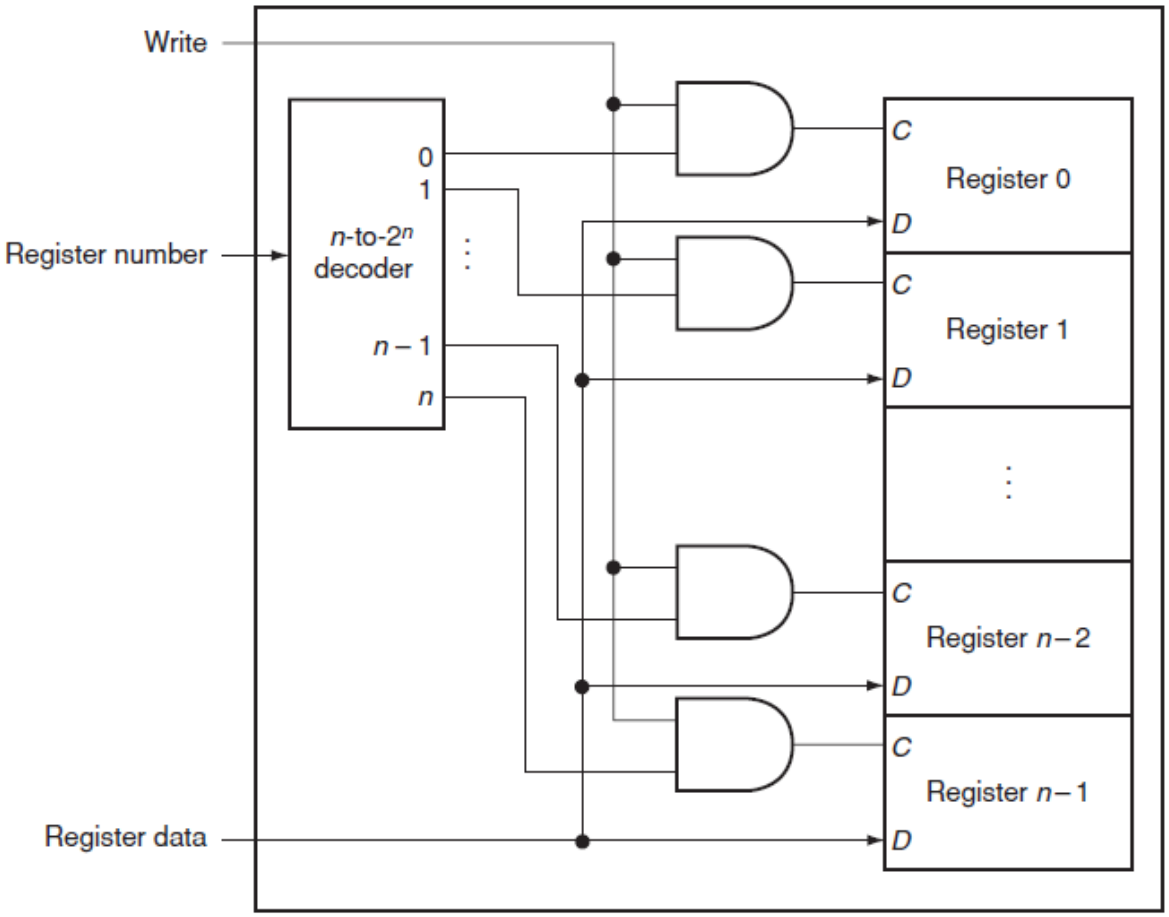
The Register File



For MIPS, n = 32

Register read number controls which input the Mux allows to pass through

The Register File



For MIPS, n = 32

Writes to register 0 have no effect

5-to-32 decoder selects register to be written

Data input supplies value to write into selected register