CS253Architectures II

Lecture 11

Memory Hierarchy

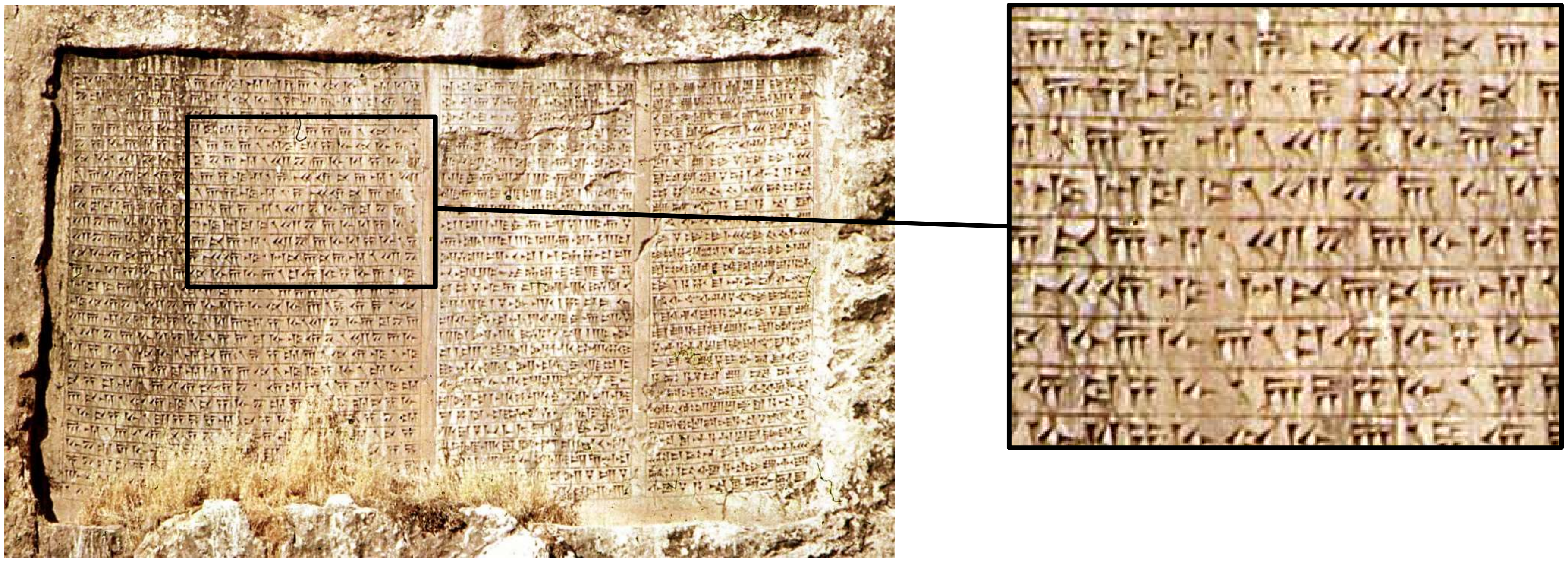
Dynamic and Static Ram

Charles Markham

Overview

Memory architecture describes the methods used to implement electronic computer data storage in a manner that is a combination of the fastest, most reliable, most durable, and least expensive way to store and retrieve information.

Memory a means of 'remembering '



Cuneiform script (Sumarian, Turkey) 3100BC (5000years)

Stone Tablet
5000BC
100's bytes

Papyrus
3000BC
1K bytes

Vellum (calfskin)
800AD
10'sKbytes

Paper book
1455AD
10's bytes

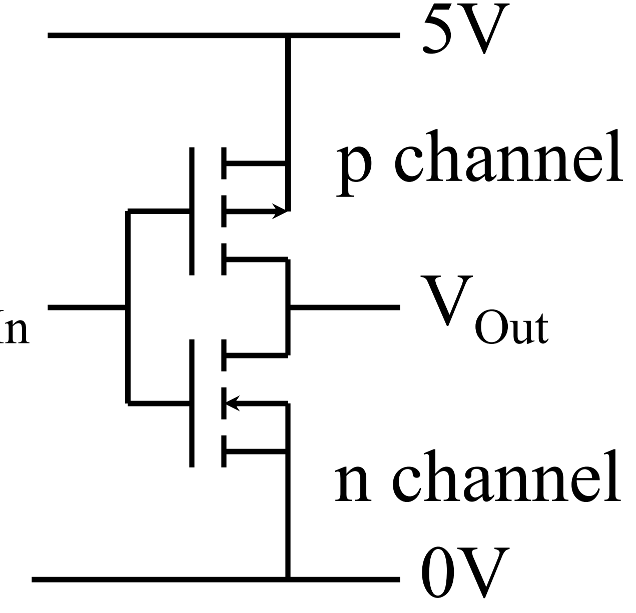
LP
40mins Audio

Tape
660 kBytes C90

VHS
1976
6 GB 200min

Toward modern memory

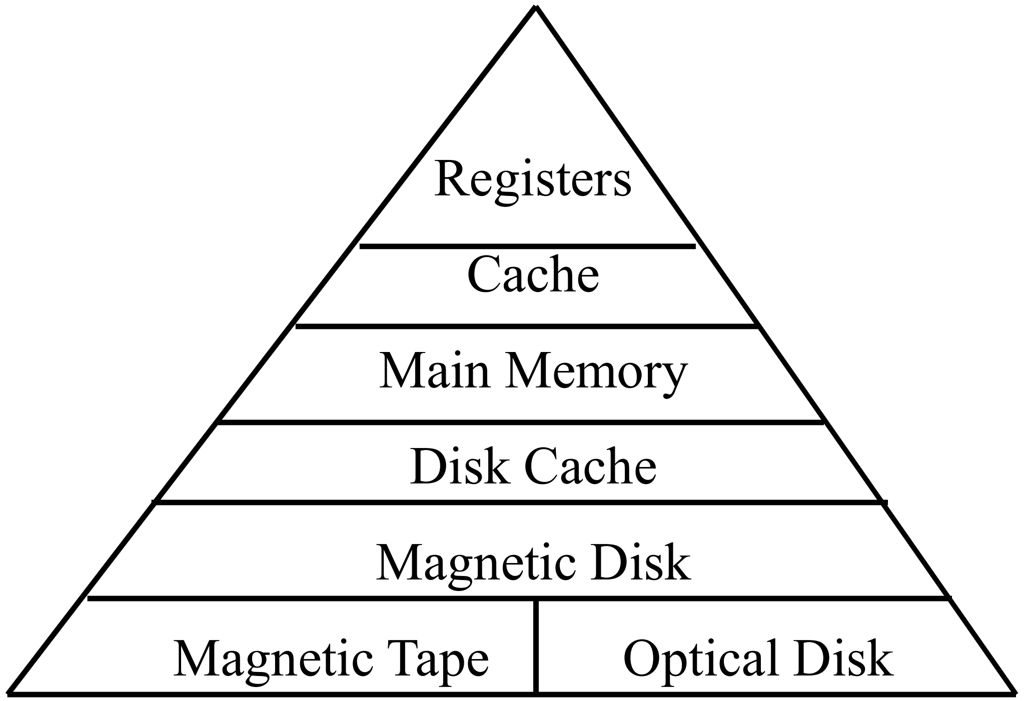
CMOS Inverters



Note: there is only dissipation during switching

CMOS-Complementary Metal Oxide Semiconductor

Memory Hierarchy



Location, Capacity, Access Method, Speed, Volatiliy

Memory Hierarchy

	Location	Capacity	Access	Speed	Volatility
Registers	On CPU chip	Bytes	Instruction	<3 clock cycles	Yes
Cache	On or off CPU chip	M-bytes	Addressing	10nS	Yes
Main memory	Motherboard	G-bytes	Addressing	80nS	Yes
Disk	In PC Case	T-bytes	Interface	10mS	No
DVD	In/Out PC case	G-bytes	Interface	100mS	No

Note: DVD and Disk can transfer data fast (Mbits/sec), the time shown is the latency (time taken to read the first byte).

Main Memory

Static Ram

Construction based on Flip Flops

Each memory cell uses up a lot of real estate

Memory is volatile, but does not need refreshing

Much faster than Dynamic Ram, used for caches, 10nS

Dynamic Ram

Construction based on charge stored on the gate of a FET.

Much higher densities possible (larger memory)

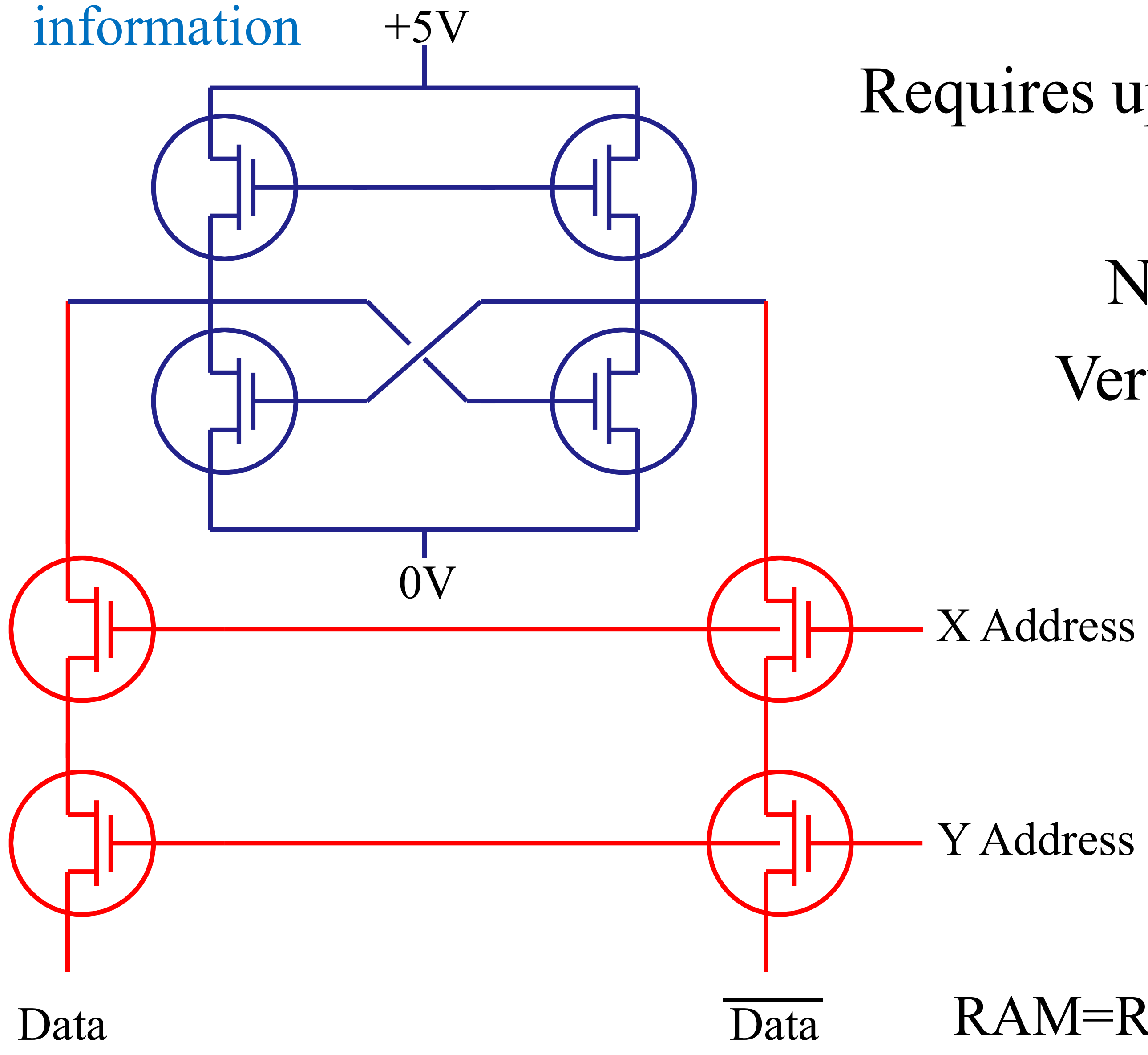
Lower Cost

But requires extra refresh control signal.

Typically 60nS, no dissipation when not accessed.

Static RAM Cell

Stores 1 bit of information



Requires up to 8 FETs per bit

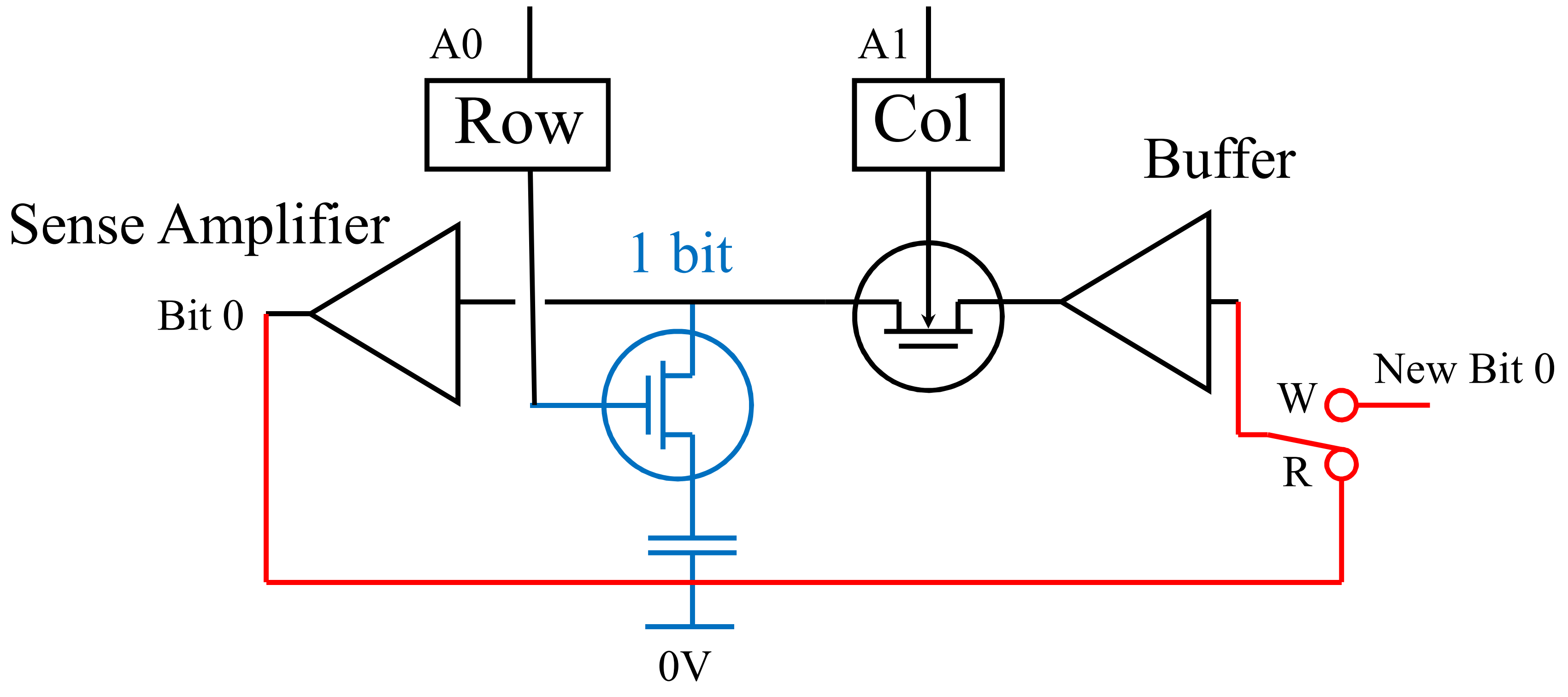
Volatile

No-refresh

Very fast 10nS

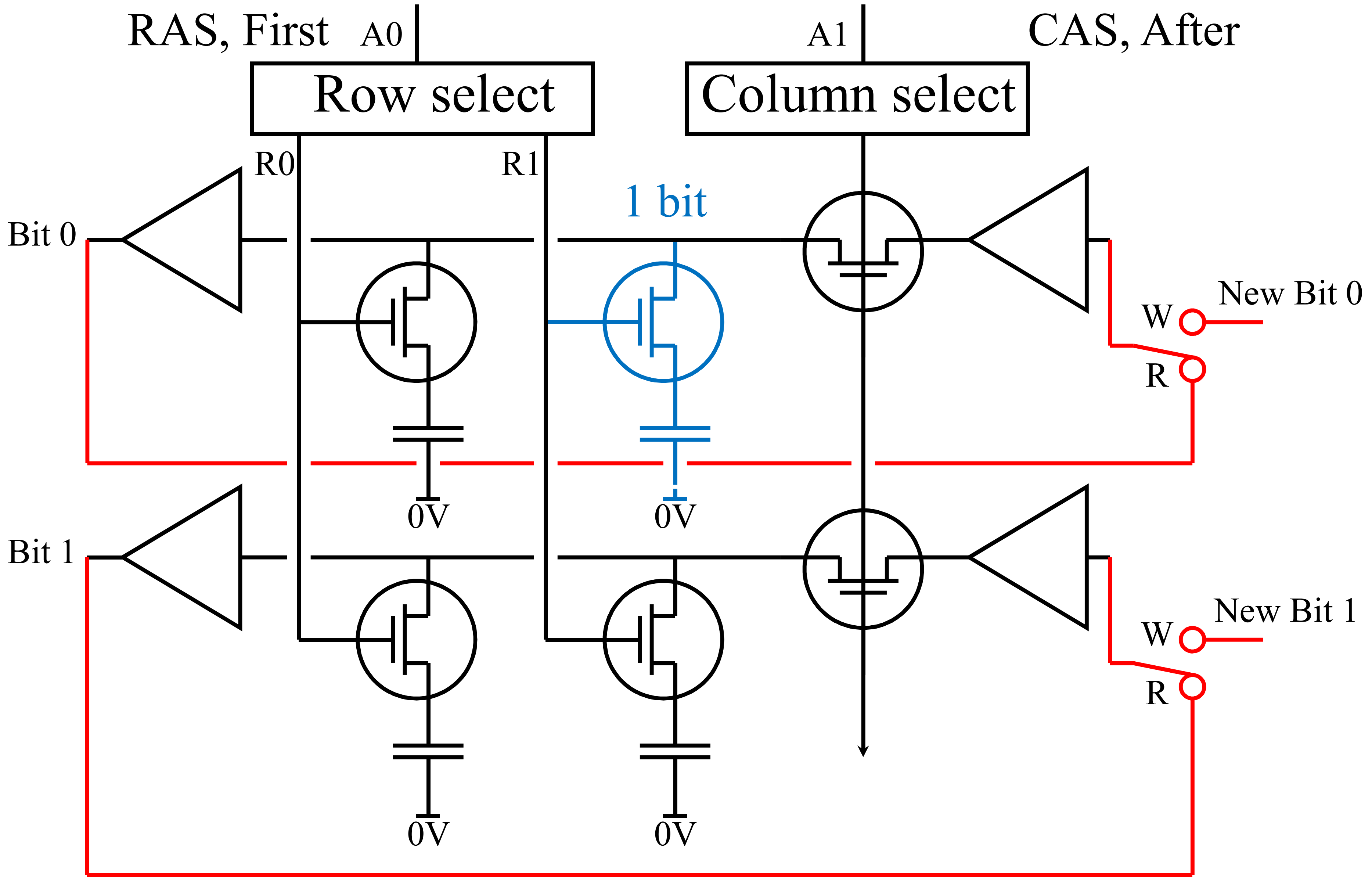
RAM=Random Access Memory

Simplified Dynamic Memory 1 bit

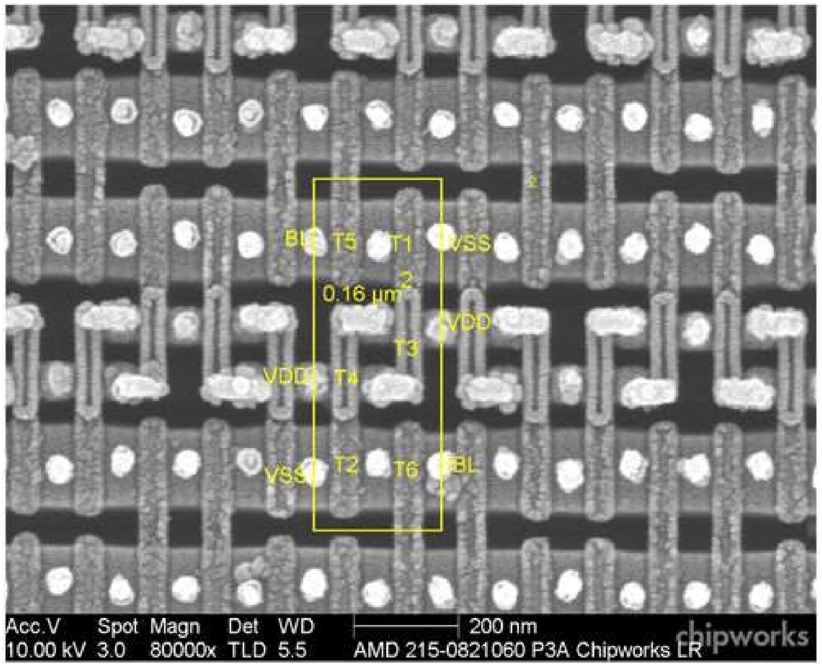


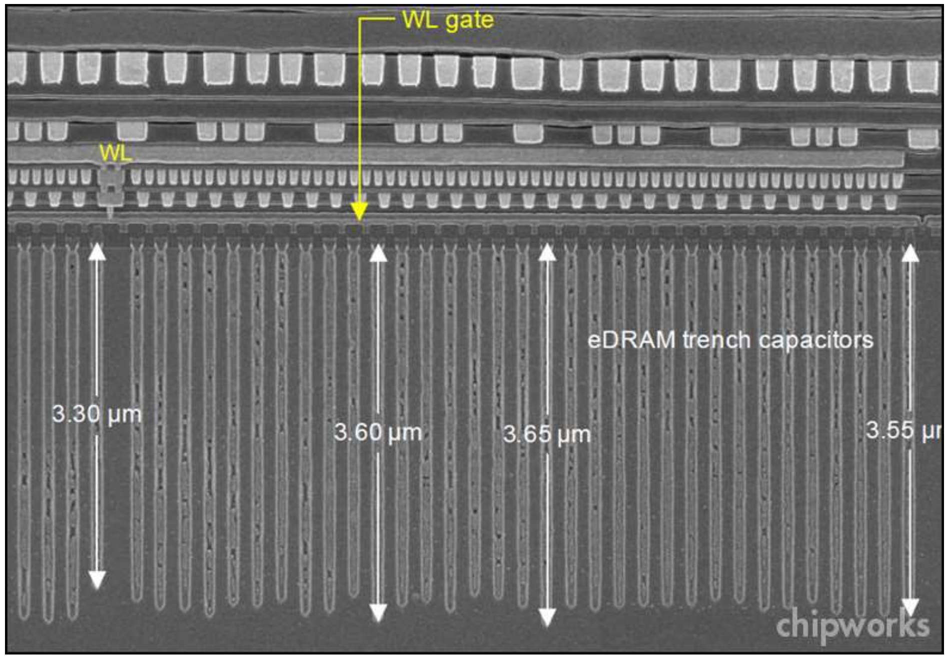
Information stored as charge on capacitor.
Needs refreshing (rereading) to keep charge on capacitor.
Requires apx 1 FET per bit.
Volatile
Slower 100nS to read or write

Simplified Dynamic Memory 4 bit



Compare SRAM and DRAM





+ 6T SRAM 200nm x 500nm

1mm^2=10^12/200*500=10M
Apx. 2Mbytes

DRAM 3.6um x 0.6um

1mm^2=10^12/360*6=46M
Apx. 6Mbytes

+ www.chipworks.com

For the same silicon DRAM gives x3 the memory.

Dynamic Ram Cycle

A row is selected

Data is read out of each cell

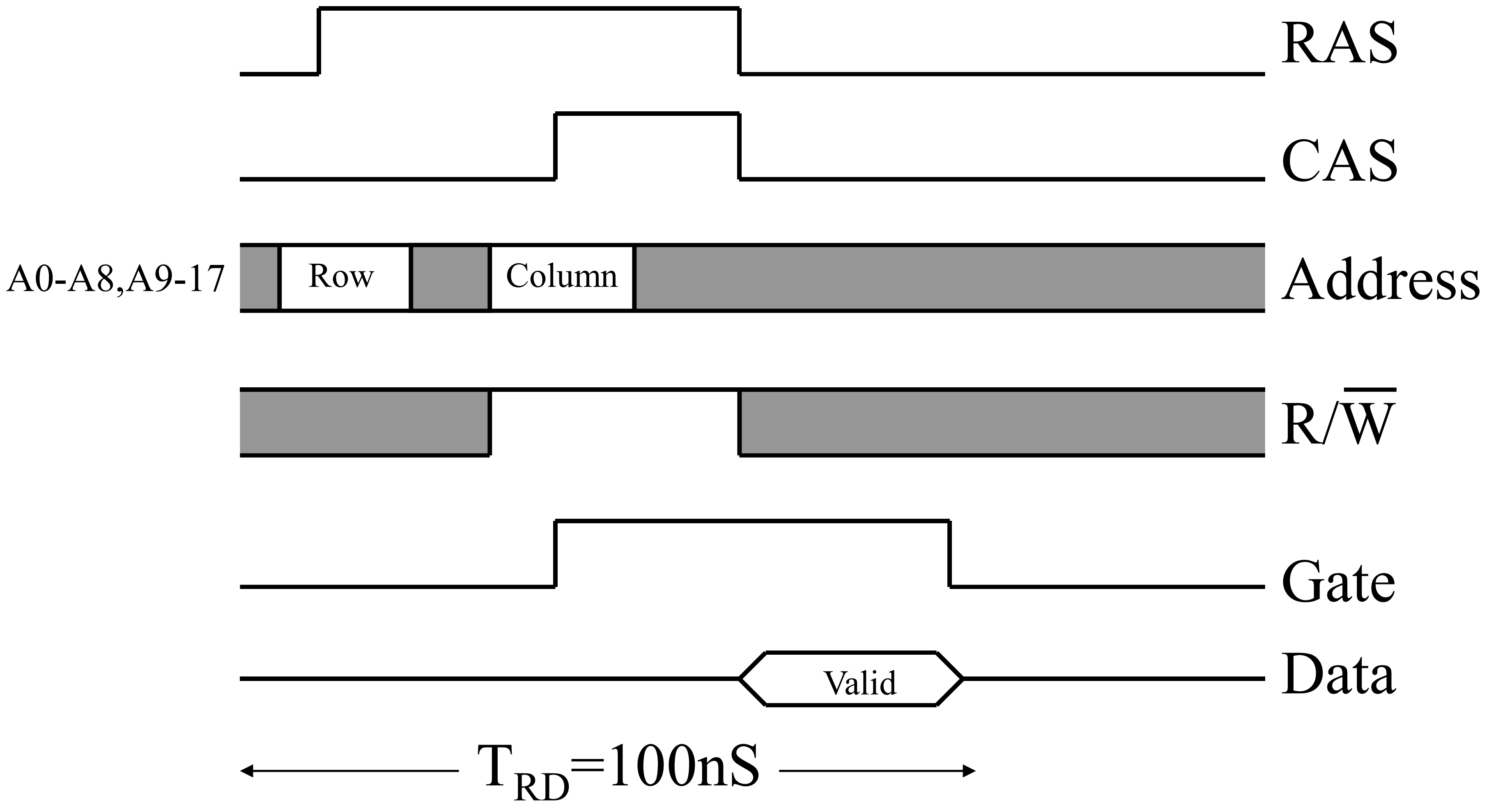
If RAM is in read mode the column line is activated and the data written back to the memory cell.

If RAM is in write mode the column line is activated and new data is written to the memory cell.

Each memory cell must be re-written every 2mS (refreshed) so that charge is not lost from the capacitor in the memory cell, the RAS and CAS lines are used to achieve this.

RAS=Row Address Select, CAS=Column Address Select

Timing Diagram D-Ram (Read)



Dynamic Ram, Observations

To access a D-Ram a uP must

Split the address in two halves.

Select Read/Write.

Refresh each row every 2mS

Solution: Dedicated chips are available to support the uP with this task.

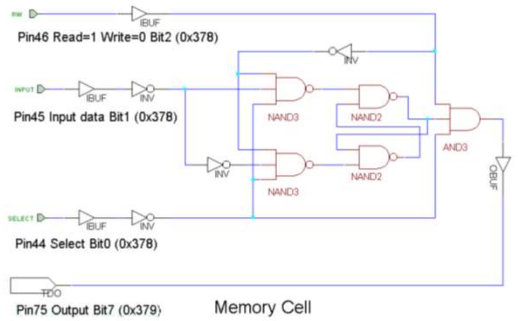
Dynamic ram precharges a cell before it can be re-read, speed of ram is thus the total cycle time T_{RD} which is about twice the access time (Typically 70nS).

Due to low speed uP has to introduce wait states when accessing dram.

Very high density available e.g. 16Mbyte per device.

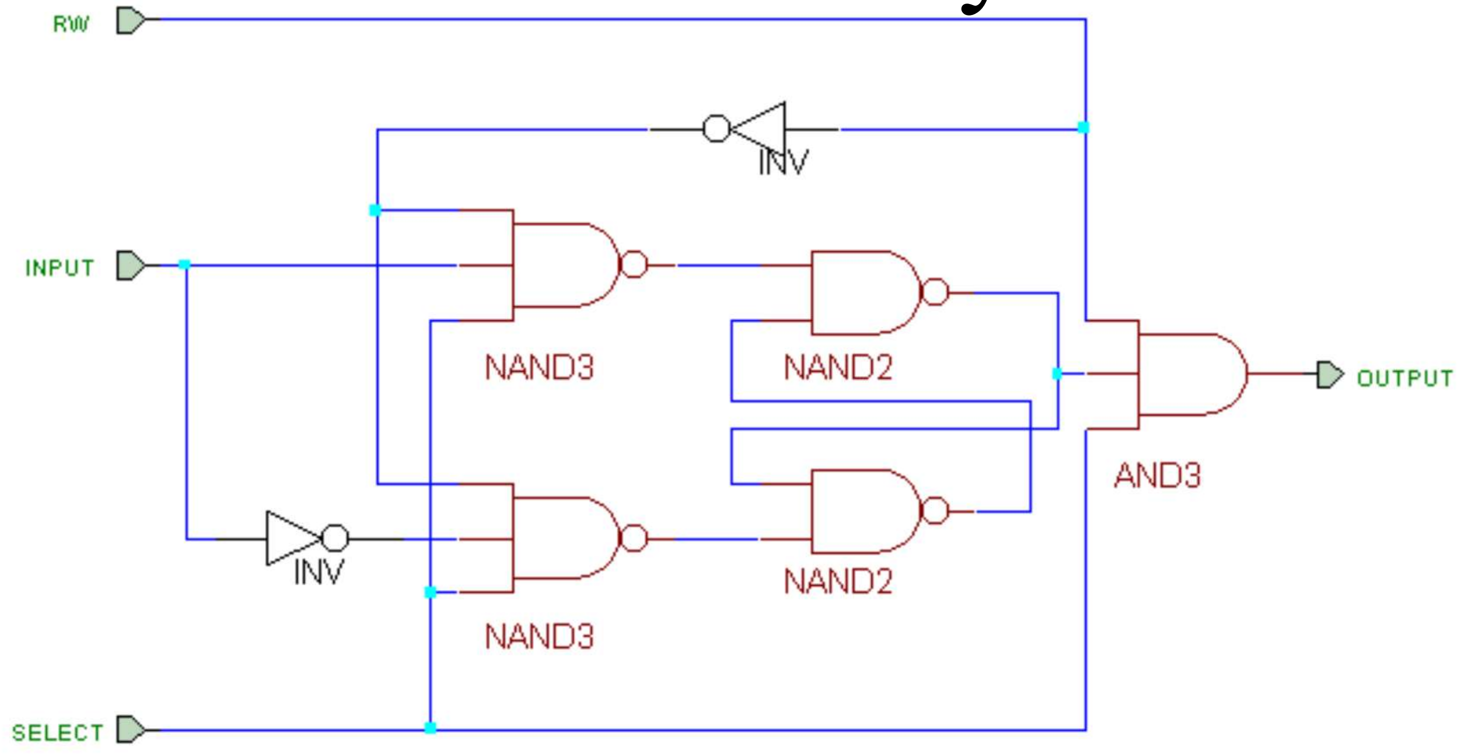
Low cost.

Static Memory (realised with gates)

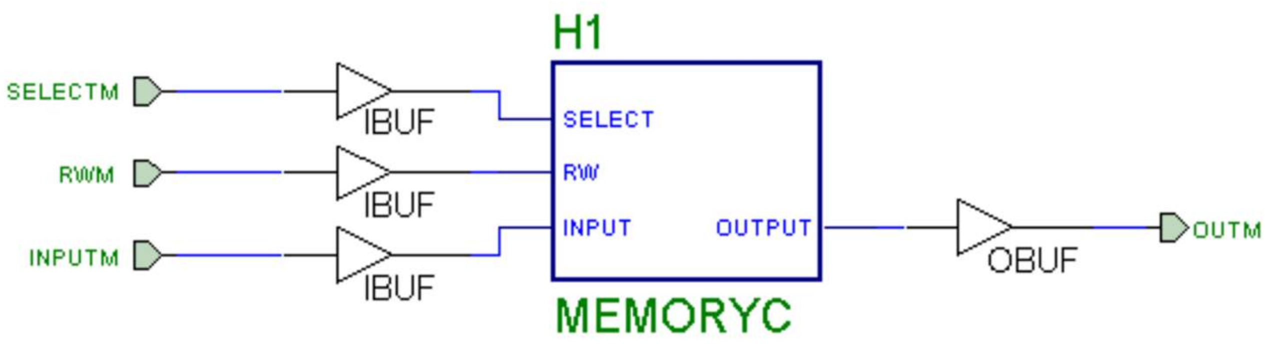

RW
IBUF
Pin46 Read=1 Write=0 Bit2 (0x378)
INV
INPUT
IBUF
INV
Pin45 Input data Bit1 (0x378)
NAND3
NAND2
AND3
INV
NAND3
NAND2
OBUF
SELECT
IBUF
INV
Pin44 Select Bit0 (0x378)
TDO
Pin75 Output Bit7 (0x379)

Memory Cell

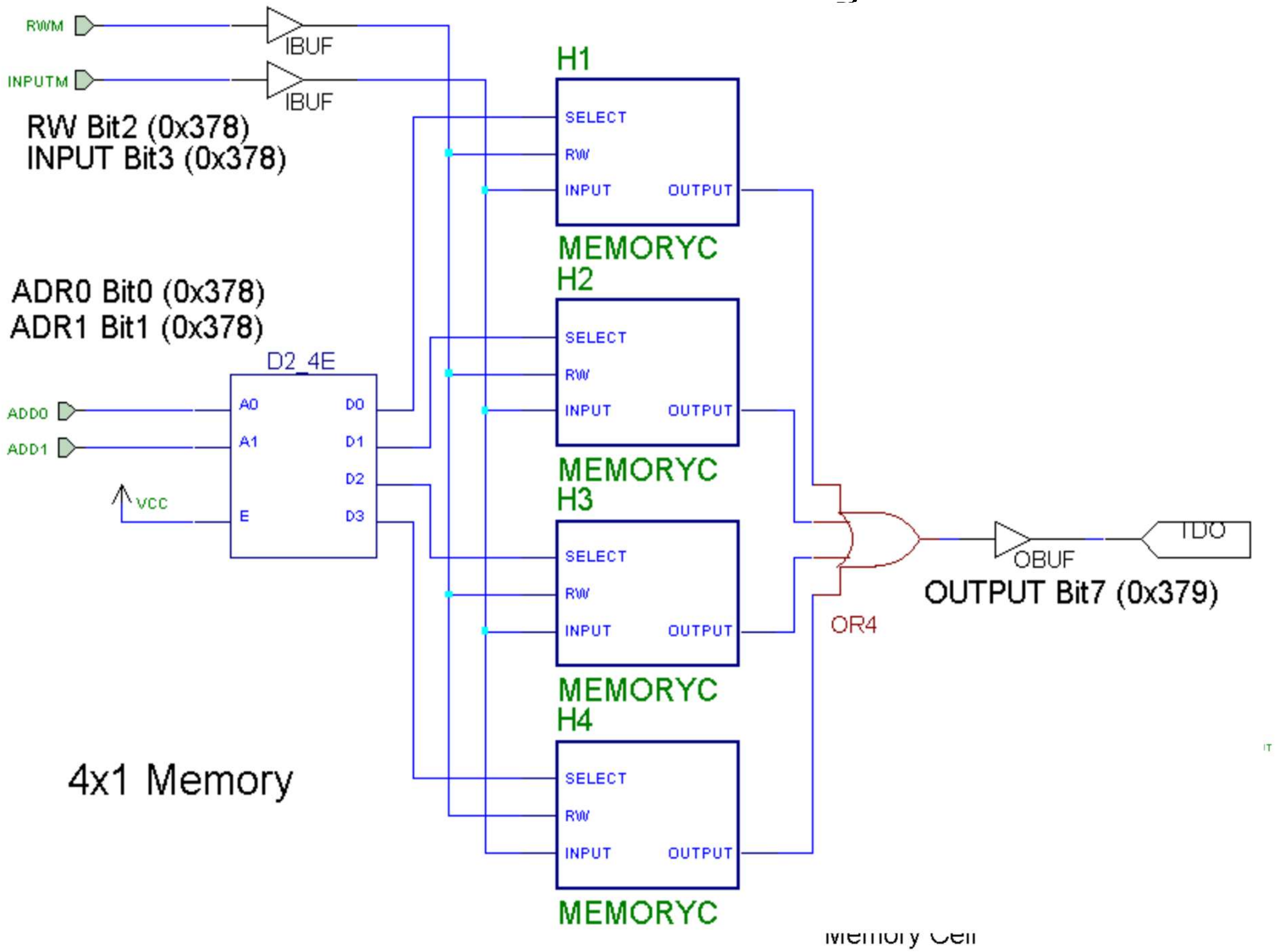
Static Memory



Memory Cell



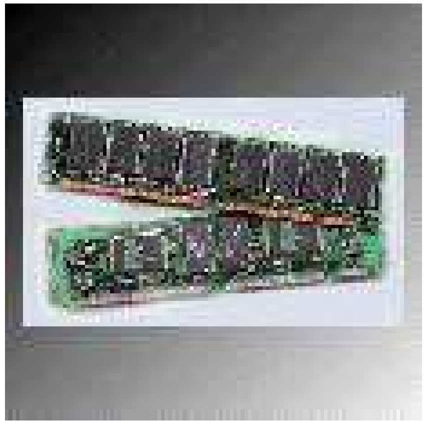
Static Memory



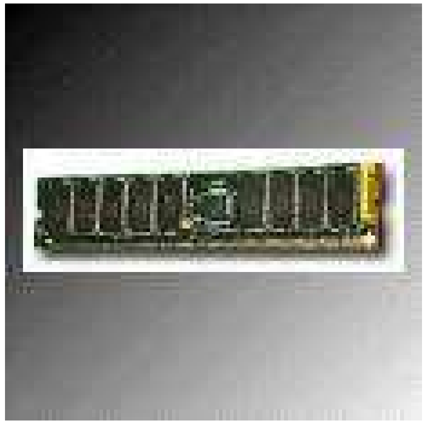
DIMS, SIMS, DIPS

DIP: Dual in line package

SIMMs, or Single Inline Memory Modules



DIMMs, Double Inline Memory Modules



CS253Architectures II

Lecture 12

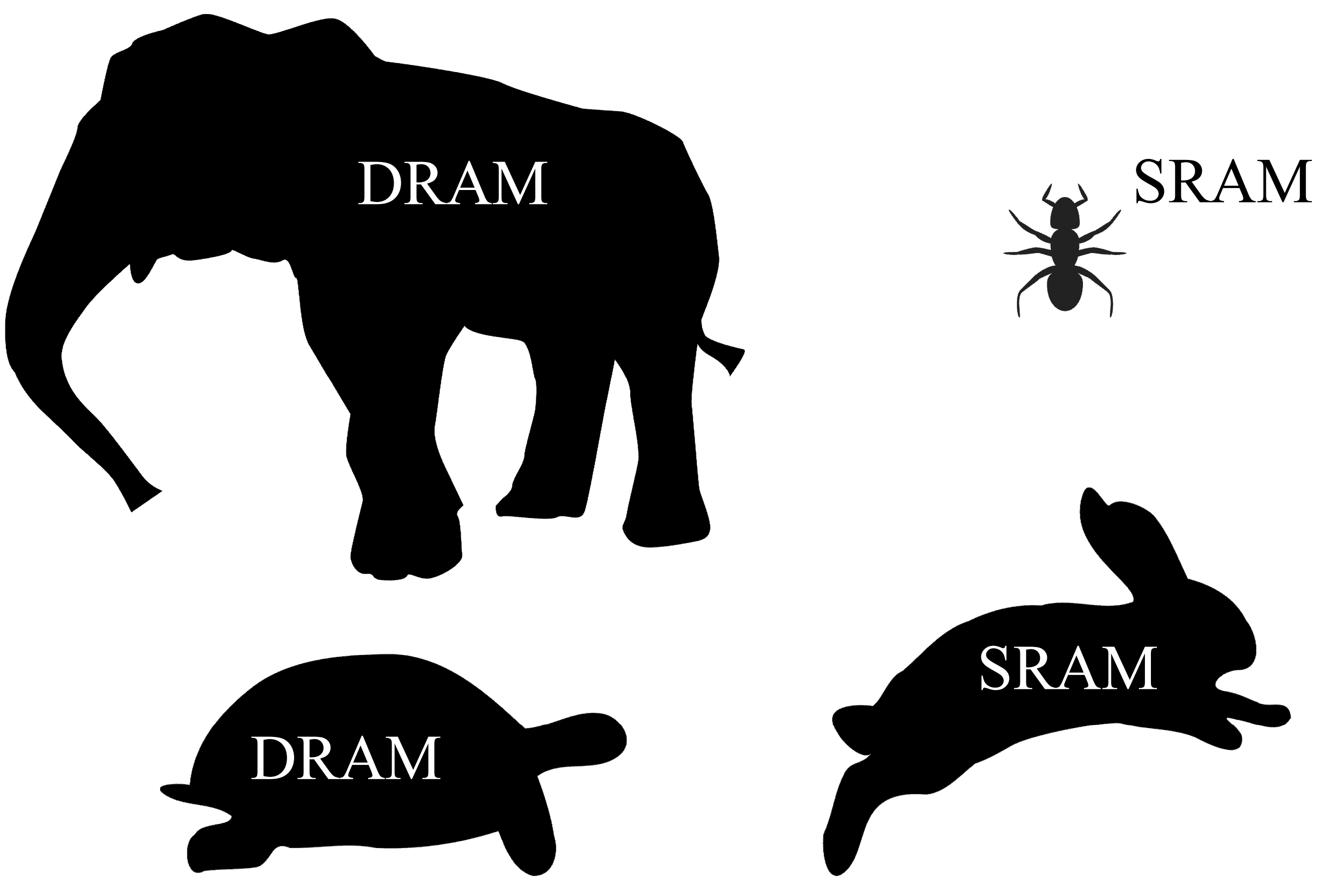
Memory Management

Cache strategies

Charles Markham

Improving memory

Is there a way of combining the Speed of SRAM with the high storage capacity of DRAM?



Locality of reference

Temporal locality (locality of time): If an item is referenced, it will tend to be referenced again soon.

Spatial locality (locality in space): If an item is referenced, nearby items will tend to referenced soon.

Another look at Hello world

Bytes stored in data segment, spatial locality.

Code in loops temporal (and spatial) locality.

```
                                        .MODEL   medium
                                        .STACK
0000                                    .DATA

0000    48 65 6C 6C 6F 2C    msg1       BYTE   "Hello, world.$"
        20 77 6F 72 6C 64
        2E 24

0000                                    .CODE

                                        .STARTUP

0017    BB 0000 R                       mov bx,OFFSET msg1
001A    8B 17               back:       mov dx,[bx]   ;dl=letters

001C    80 FA 24                        cmp dl,'$'
001F    74 07                           jz  done

0021    B4 02                           mov ah,02h
0023    CD 21                           int 021h

0025    43                              inc bx
0026    EB F2                           jmp back

0028    90                  done:       nop

                                        .EXIT

                                        END
```

Cache

The code and data currently being used by the uP are copied from D-RAM (128Mbyte, 180nS) to S-RAM (256Kbyte, 45nS) (P100).

Intel i7 Level 2 Cache 8Mbyte cache, typically 8 Gbytes DRAM

When the uP sends out an address to the cache controller it checks to see if the data is in cache or main memory.

If it is in cache the data is sent no and there are no waits to the uP.

If it is not in the cache the data is read from main memory and sent to the uP (slowly) a copy is kept on file in the cache for future use.

The percentage of S-RAM to (DRAM+SRAM) reads is known as the *hit rate.* The hit rate is a figure of merit for the effectiveness of the cache. Hit rates on a PC are typically 90%.

Typical Cache Parameters

Block (line) size	4-128 bytes
Hit Time	1-4 Clock cycles (1 is best)
Miss Penalty	8-32 clock cycles
Miss rate	1%-20%
Cache size	32KB-64MB

Cache

The code and data currently being used by the uP are copied from D-RAM (8Gbyte, 180nS) to S-RAM (8Mbyte, 45nS) (Intel i7).

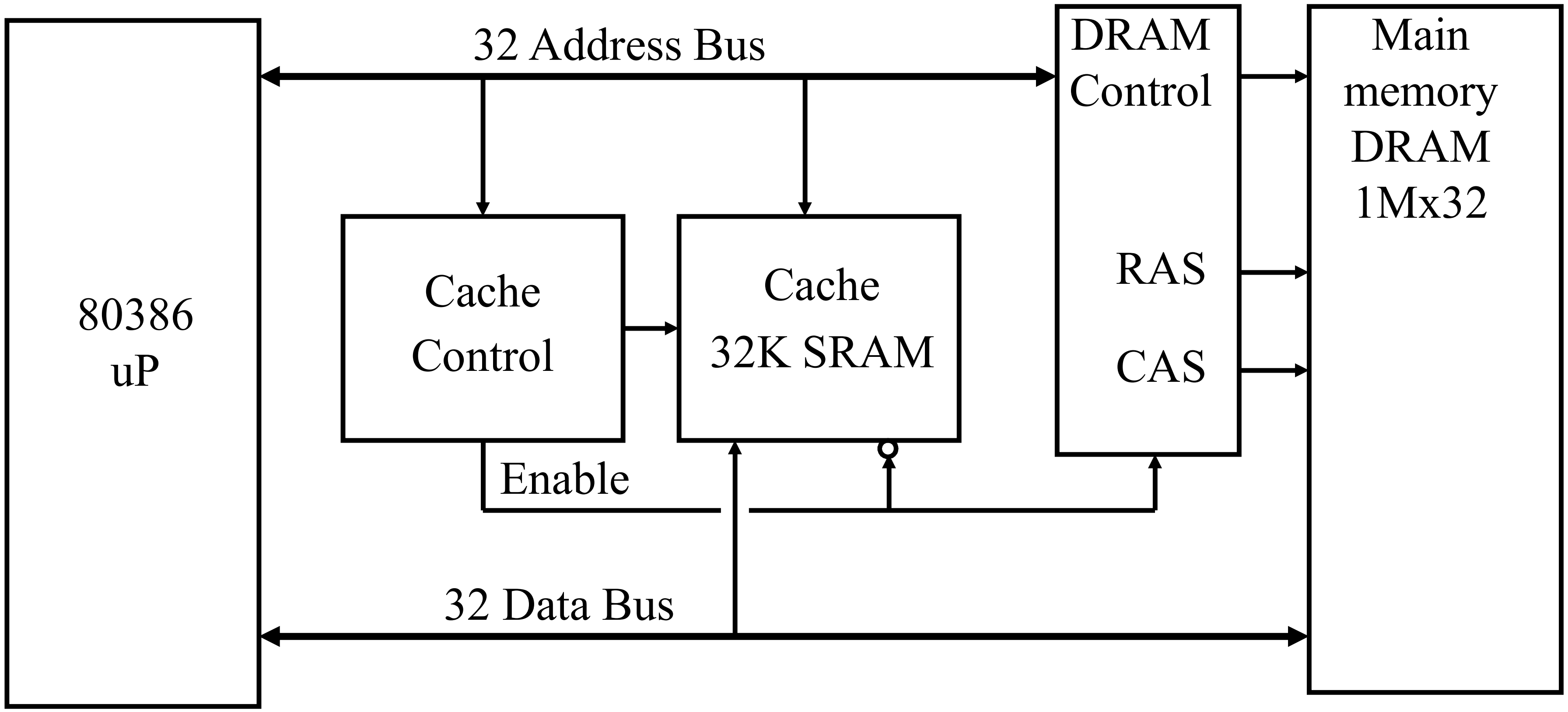
When the uP sends out an address to the cache controller it checks to see if the data is in cache or main memory.

If it is in cache the data is sent and there are no waits to the uP.

If it is not in the cache the data is read from main memory and sent to the uP (slowly) a copy is kept on file in the cache for future use.

The percentage of S-RAM to (DRAM+SRAM) reads is known as the *hit rate.* The hit rate is a figure of merit for the effectiveness of the cache. Hit rates on a PC are typically 90%.

Typical PC RAM system, x86



Typical Cache Parameters

Block (line) size	4-128 bytes
Hit Time	1-4 Clock cycles (1 is best)
Miss Penalty	8-32 clock cycles
Miss rate	1%-20%
Cache size	1KB-1MB

Organisation of the cache

The cache controller contains a cache directory.

Direct Mapped Cache: The cache is organised to contain a copy of a single page of memory. If processor switches rapidly between pages the effect is known as *thrashing* this will slow the system down.

Two-Way Set Associative: Uses two separate caches to allow fast switching between pages and reduce chance of thrashing.

Fully Associative: Each d-ram address and associated data is stored in cache memory as it is used. Could be slow since you need to look through the directory. How do you discard old data in cache?

Direct Mapped Cache

An 80386 has a 32bit address bus and can access 2^{32}=4GBytes of data.

The data bus is also 32 bits wide, each group of 4 bytes is called a line.

The 80386 cache contains 8196 lines a total of 32K memory.

The cache groups lines into blocks of 8 lines, each 32bytes long, 256 bytes.

The cache controller treats the 4GByte memory as

2^{32}/32768=131072 pages of 32Kbytes each.

A particular line from main memory will always map to the same line in cache.

The cache directory will store the page location (main memory) for each line in cache. This information is known as the tag.

Direct Mapped Cache

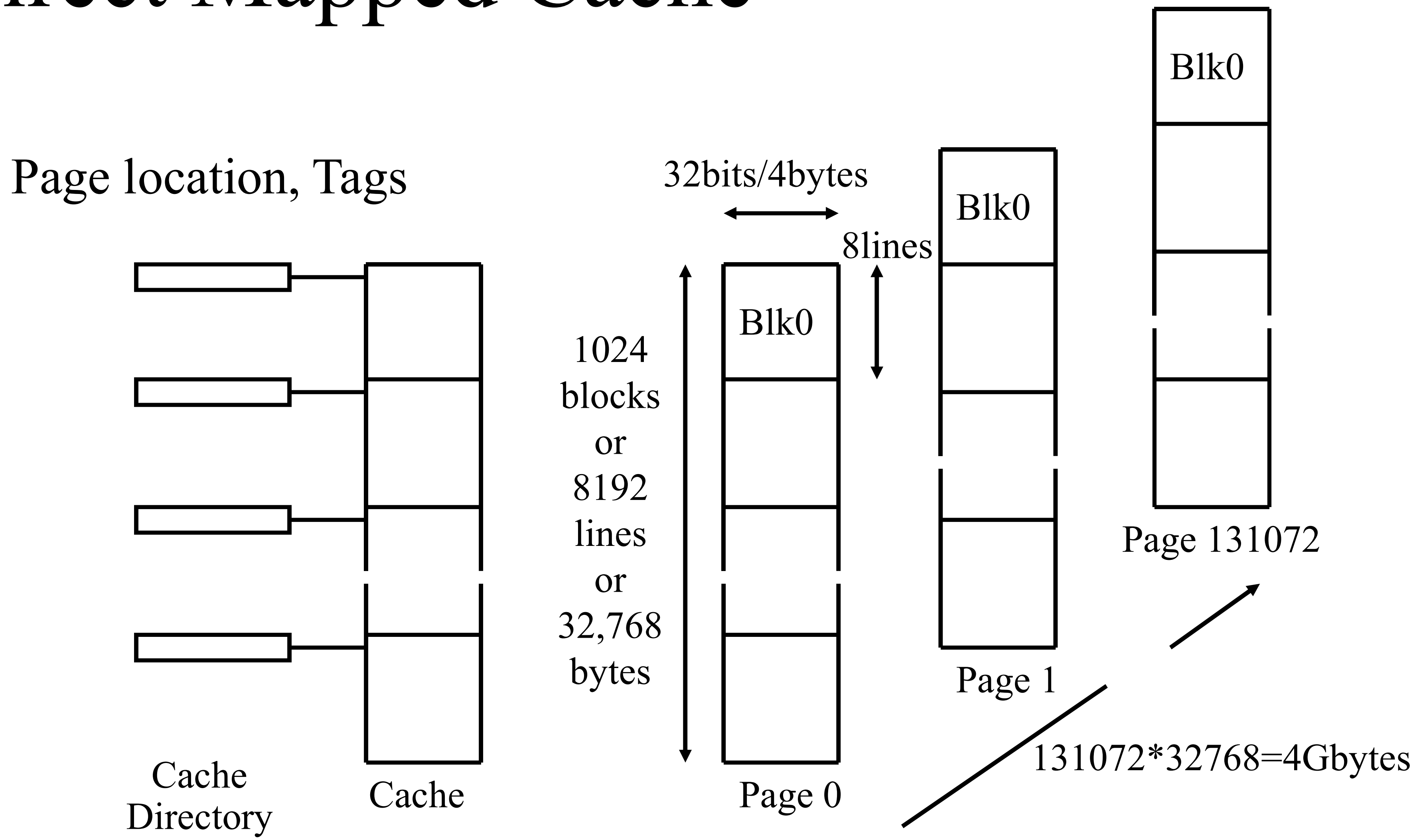
Accessing memory

The uP outputs a 32bit address the upper 17bits are compared with the cache directory entry, if present and the tag bit valid data is read from the cache (a hit).

If it is not present or the tag bit not-valid then the cache controller access D-RAM for the data, a copy is stored in cache for subsequent use.

A15-A32:	Set the page in main memory
A5-A14:	Set the block number in cache
A2-A4:	Set the line number in the block
A0-A1:	Specific bytes in the line

Direct Mapped Cache



Cache directory stores **page numbers** and **validity bit**

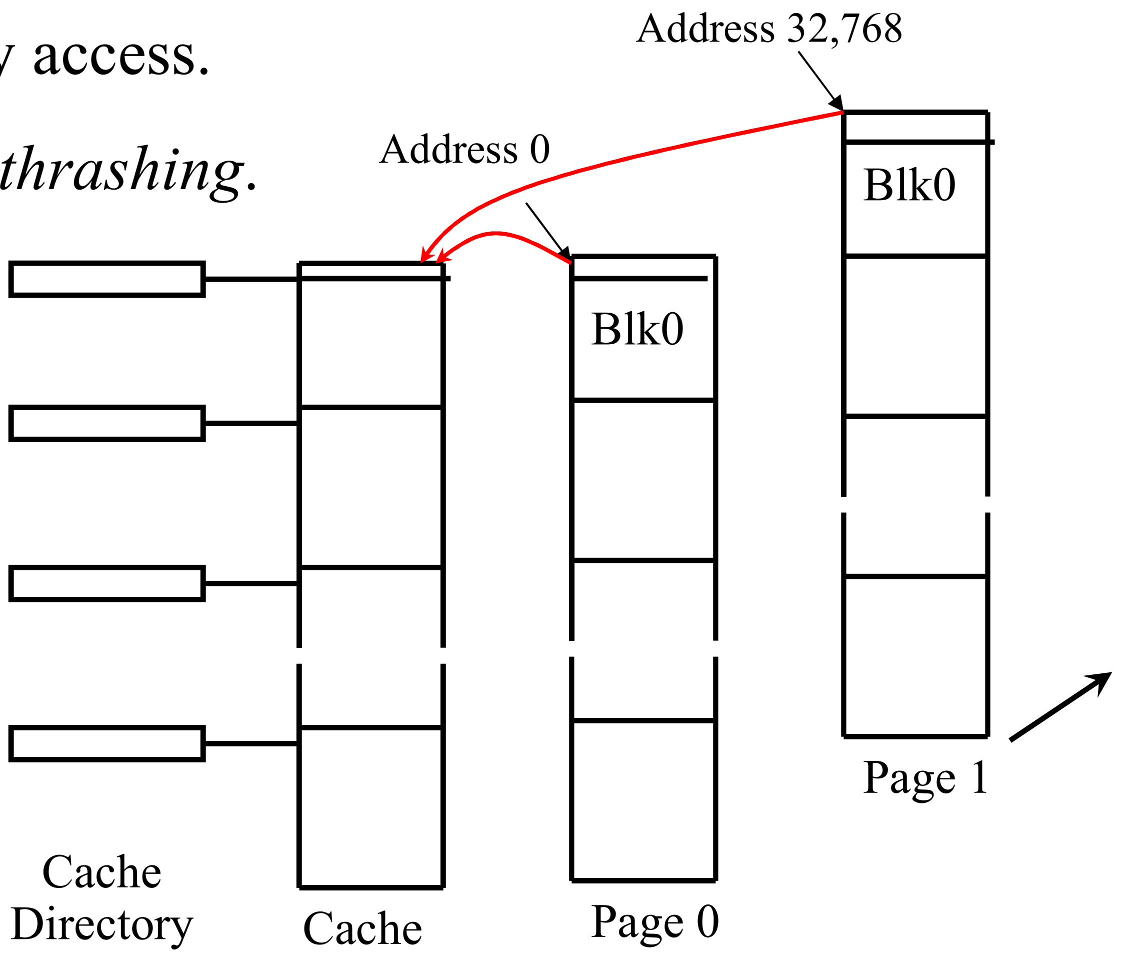
Thrashing

If a program accesses page 0 block X and a then page 1 block X as part of a repeating loop. Each memory access will result in the DRAM access and an update of the cache.

This will delay memory access.

The effect is known as *thrashing*.

For example thrashing could occur if a program causes address 0 then address 32,768 to be called repeatedly. This could either be due to the data or to the code (both use the same memory in a von



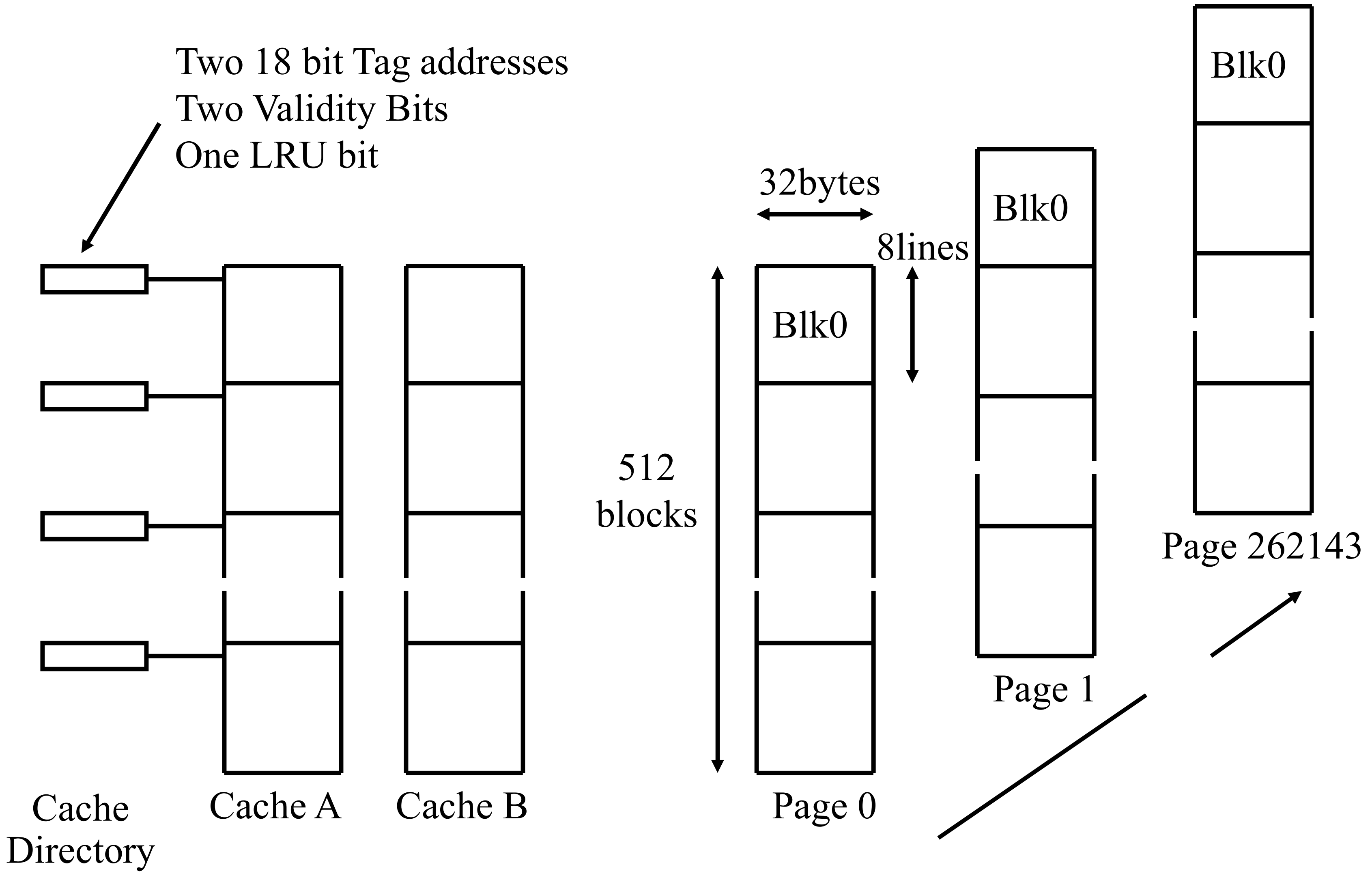
Set Associative

On a PC the thrashing problem can be solved by splitting the cache into two.

Each cache has its own directory.

An additional bit in the tag directory identifies which cache contains the most up to date version in the case both caches contain the same page. (LRU least recently used bit).

Two-Way Set Associative



Sometimes Tags are stored in special TAG ram

Fully Associative

A line (4bytes) can be written to any location in cache.

Each tag in the cache directory must be 30 bits.

This cache can hold the same line from many different pages of memory.

Not used on PC’s (1995).

Problems:

1/ you have to search through the cache directory to find your data.

2/ How do you discard old stuff when the cache is full?

Random discard : Uniform

Least Recently Used: Tricky to keep count of number of reads

What causes a cache to miss?

Compulsory: First read of each block must be from D-RAM,

cold start miss.

Capacity: If the number of blocks in regular use exceeds the cache size, then thrashing can occur.

What effect does a cache have?

Amdahl's Law: The improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used.

$$Speedup = \frac{1}{(1-p) + \frac{p}{s}}$$

s is the speedup due to the improved resources;
p is the proportion of execution time that the part benefiting from improved resources originally occupied.

The addition of a cache improves the speed of memory by 10 (from 100nS to 10n) so s=10, a typical program has a hit rate of 95% so p=0.95.

$$Speedup = \frac{1}{(1-0.95) + \frac{0.95}{10}} = 6.89$$

The addition of the cache makes memory x6.89 faster than DRAM

Estimating Effective Access Time

- Memory access time = 10 nanoseconds cache
- 5% of the time the miss rate
- 100nS dynamic ram spee

$$\text{EAT} = (1 - p) \times 10 + p\,(100)$$

$$\approx 10(1-P) + 100P$$

$$= 9.5 + 5$$

$$\approx 14.5\text{nS}$$

100ns/14.5ns=6.89   Speedup (Same as Amdahl's Law slide earlier)

Write Policies

D-RAM and Cache must always be the same.

The DMA could cause problems with cache.

Write through:	Every write to cache is also made to main memory.
	Write is slowed down to D-RAM speed.
	Reading is at cache speed.
Write back:	Data is written to the cache only.
	When the block is replaced the old block is transferred to main memory.
	Uses less memory

Multilevel caches

On modern PC's there are two or more levels of cache.

Primary (Level 1 cache) is built onto the uP chip, typically 1K to 32K.

These caches can have zero wait states.

e.g. mov ax,[bx] is done in the minimum number of clock cycles.

Secondary 2Mbyte, 500MHz

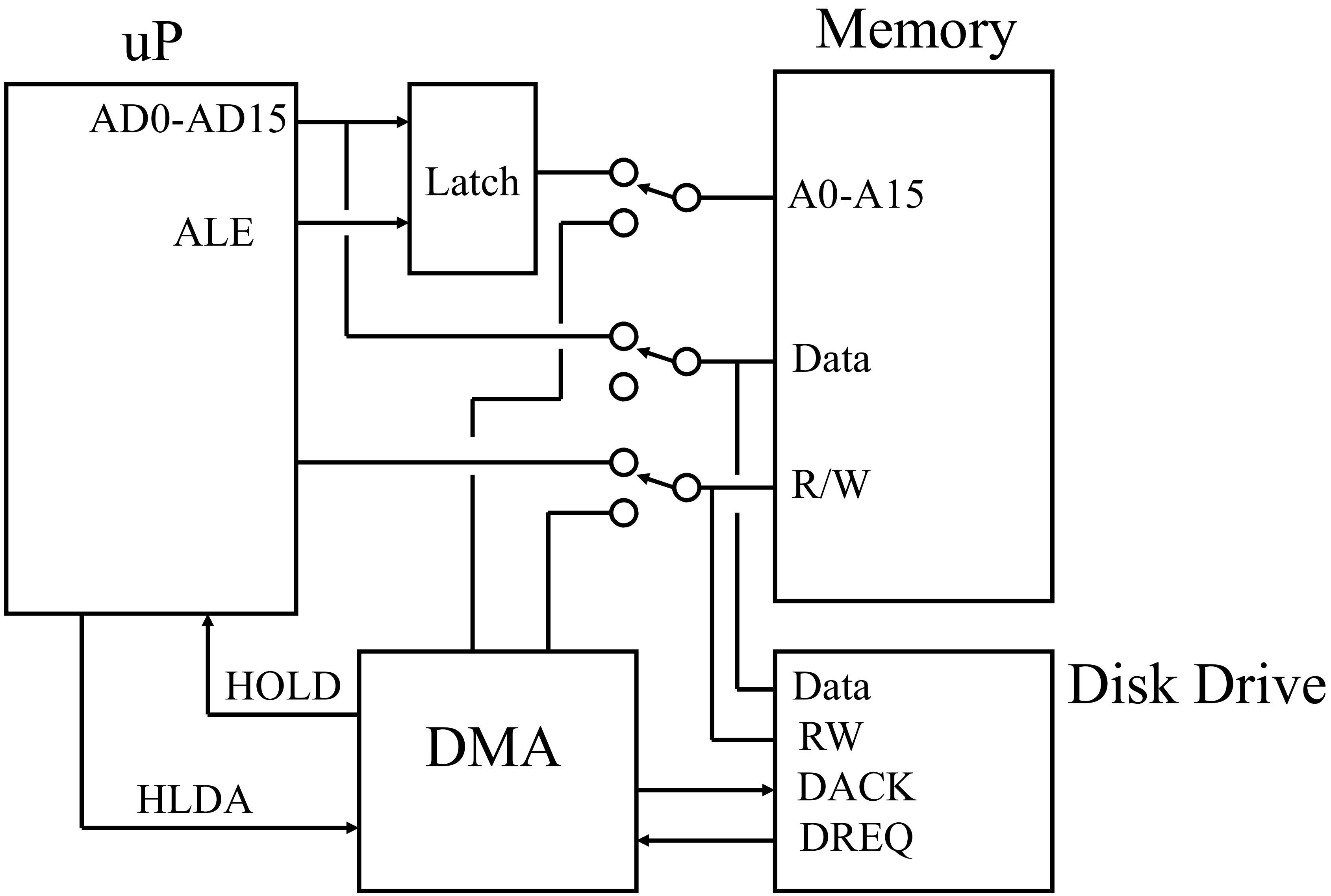
Main memory up to 64Gbytes

DMA on 8086

Many devices are able to write directly to memory. If this can be done it frees up valuable CPU time. The techniques is known as DMA or Direct Memory Access.

DMA is used on devices such as frame grabbers, and sound cards.

DMA Controllers



DMA Sequence of events

DMA controller connects uP to memory via a set of switches.

To read the disk a series of commands are sent to the smart disk controller to obtain the data to be placed in main memory. When the device is ready DREQ is set high. The HOLD (hold request) is then sent to the uP. When the uP is ready HLDA (hold-acknowledge) is brought high. The DMA controller switches the data and address line over to the smart drive. DMA sends a DACK0 (DMA acknowledge) signal. The device writes to the memory. HOLD is then released and the uP regains access to the memory.

ROMS

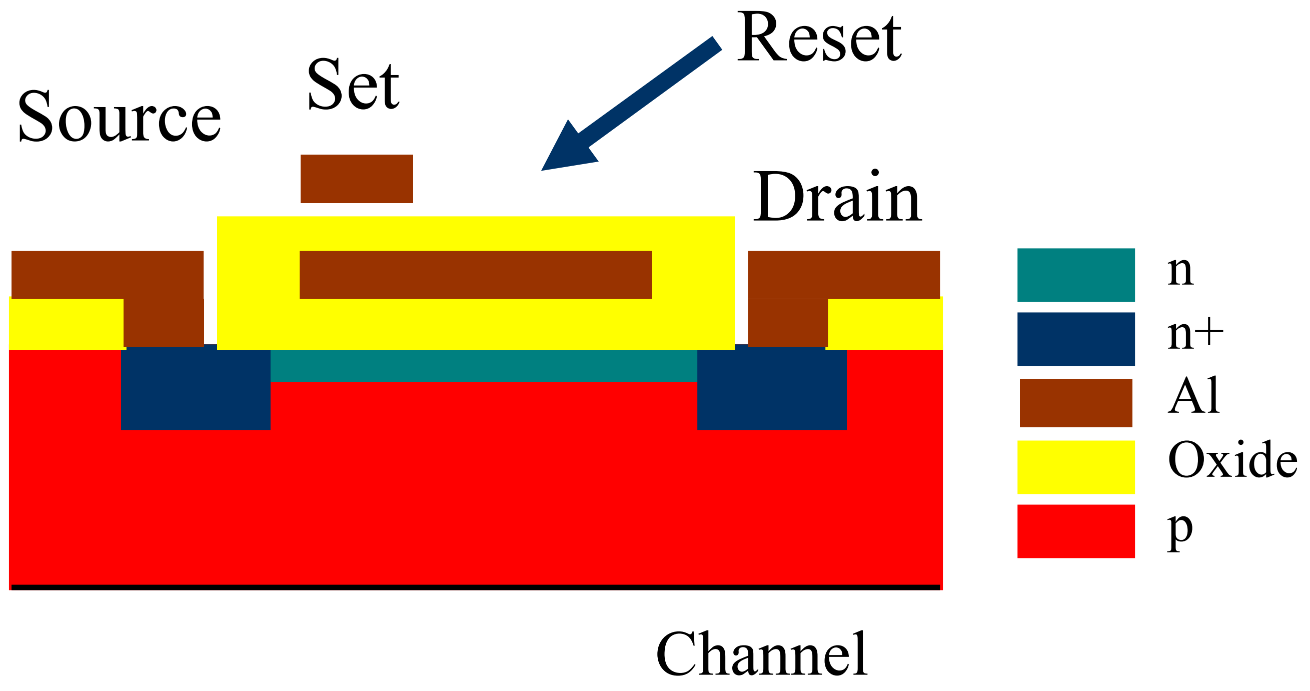
ROMS are non-volatile memories, normally they contain BIOS and Boot Sequence Code.

Mask Programmed ROM: Programmed during manufacture, cannot be altered.

PROMS:	One time programmable ROMS, programmed by blowing fusible links in the device.
EPROMS:	Programmed electrically by leaving charge on the gate of a FET. Erased using UV light that discharges the gate.
EEPROMS:	Programmed and re-programmable electrically.

ROMS are quite slow, so sometimes the contents are copied into a SRAM after the machine has been powered up.

EPROM

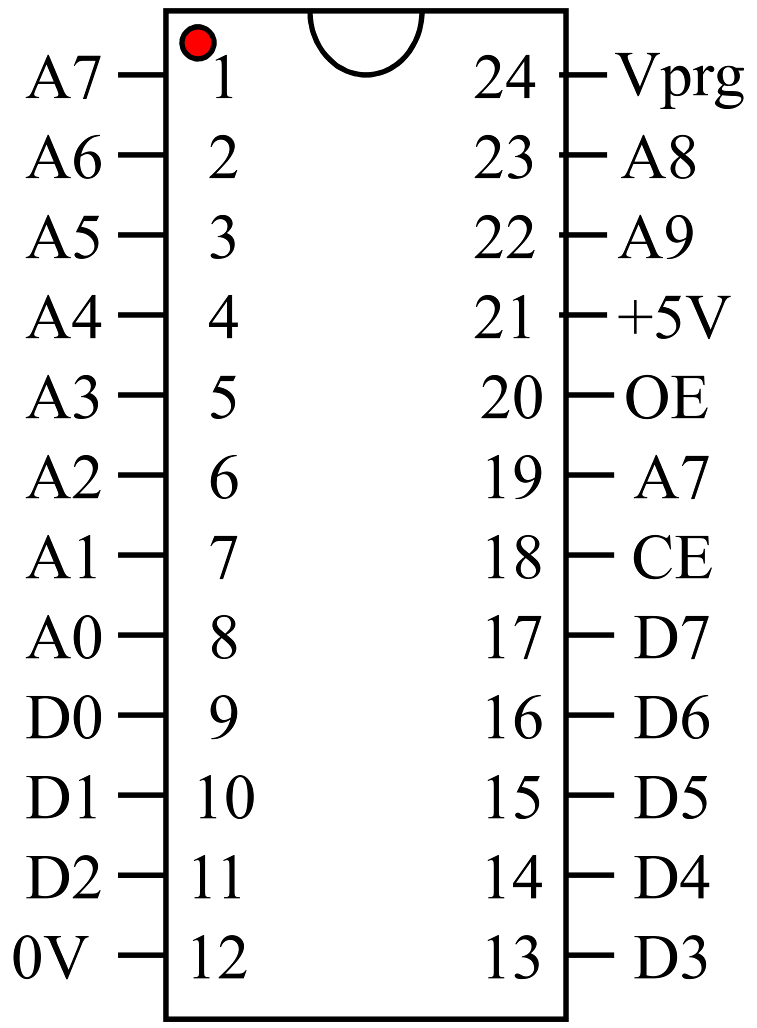


A high voltage placed on the insulated gate causes electrons to tunnel leaving the plate biased positive.

Shining light (UV) on the gate discharges the capacitor.

EEPROMS: Electrically erasable programmable read only memory.

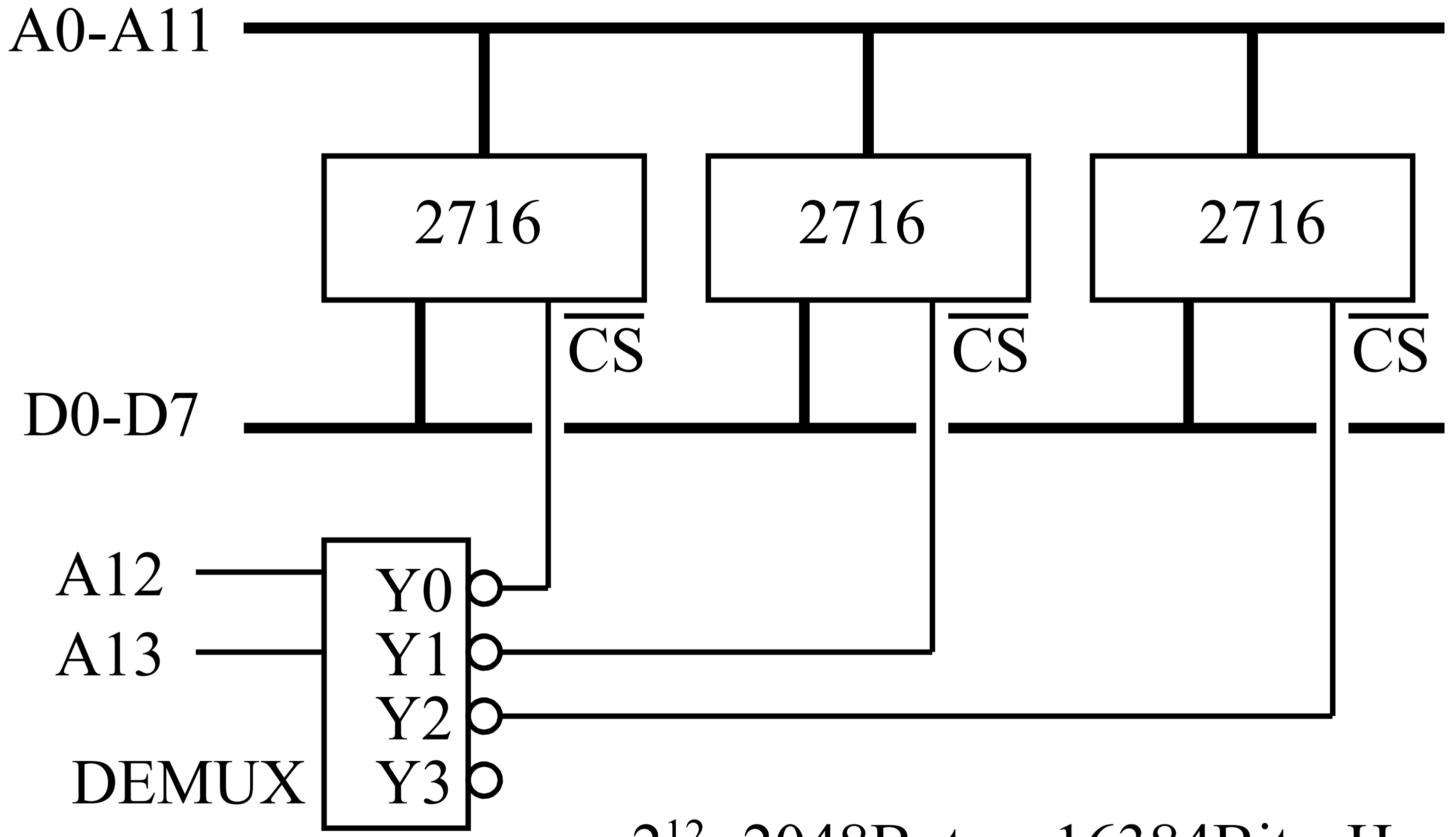
Pin-outs of a 2716 EPROM



A0-A11:	Address lines
D0-D7:	Data lines
OE:	Output enable
CS:	Chip select

Combining ROMS (or other memory)

Using the chip select CS line and a demultiplexer it is possible to to combine many memory devices in memory space.



2^{12}=2048Bytes=16384Bits, Hence 27-16

Virtual Memory

The address space mapped by the CPU is called the *virtual memory*,

on a typical PC this is 64Gbytes.

The *main memory* is the physical memory available, e.g. 4Gbytes.

A virtual memory system uses an external memory to store data.

Pages of the external memory are switched into main memory.

Same methods as caching.

Note: on older systems this could produce security problems.
Virtual memory should be cleared before use.

Virtual Memory

If there is insufficient Main memory then the Hard Disk can be used as a form of extended memory. This memory space is known a virtual memory. The miss penalty for using virtual memory can be as high as 100,000 cycles. If you open a number of applications in Windows you can fill memory. The result is that the operating system starts to use virtual memory. Things start to really slow down!