University of San Carlos | Department of
COMPUTER ENGINEERING

CpE 3202
Computer Organization & Architecture

Computer Memory (Internal)

Characteristics of Computer Memory Systems

- Location
- Capacity
- Unit of transfer
- Access method
- Performance
- Physical type
- Physical characteristics
- Organization

Location

- **CPU**
  - Registers and L1 cache
- **Internal**
  - Main
- **External**
  - Secondary

Capacity

- **Word size**
  - The natural unit of organization
  - Typically equal to the number of bits used to represent a number and to the instruction length
- **Number of words**
  - Or bytes
  - Has to do with the number of addressable units

Unit of Transfer

- **Internal**
  - Usually governed by data bus width
- **External**
  - Usually a block which is much larger than a word
- **Addressable unit**
  - Smallest location which can be uniquely addressed
  - Word internally
  - Cluster on M$ disks

Access Methods (1)

- **Sequential Access**
  - Start at the beginning and read through in order
  - Access time is highly variable – *depends on location of data and previous access*
  - e.g. tape
- **Direct Access**
  - Individual blocks have unique addresses based on physical location
  - Access is direct by jumping to vicinity plus sequential search
  - Access time is still variable – *but not as much as sequential – depends on location of data and previous access*
  - e.g. disk

Access Methods (2)

- **Random Access**
  - Individual addresses identify physical locations exactly
  - Access is direct to desired location
  - Access time is independent of location or previous access
  - e.g. RAM
- **Associative**
  - Data is retrieved by comparing a sub-part of the data with a desired mask *(location is not needed)*
  - All matches to the mask are retrieved simultaneously
  - Access time is independent of location or previous access
  - e.g. cache

Memory Hierarchy

- **Registers**
  - In CPU
- **Internal or Main memory**
  - May include one or more levels of cache
  - “RAM”
- **External memory**
  - Backing store

Performance

- **Access time**
  - Time between presenting the address and getting the valid data
- **Memory Cycle time**
  - Time may be required for the memory to “recover” before next access
  - ***Cycle time*** is access + recovery
- **Transfer Rate**
  - Rate at which data can be moved (bits/sec)

Physical Types

- **Semiconductor**
  - RAM
- **Magnetic**
  - Disk & Tape
- **Optical**
  - CD & DVD
- **Others**
  - Bubble *(stationary disk with spinning bits)*
  - Hologram *(optical, will go beneath the surface and use the volume of the recording medium for storage)*

Physical Characteristics

- **Decay**
- **Volatility**
  - Volatile *(information is lost without power)*
  - Non-volatile *(information retains until changed; no electrical power needed)*
- **Erasable**
  - Erasable
  - Non-erasable *(information can't be altered with a normal memory access cycle; must be non-volatile)*
- **Power consumption**

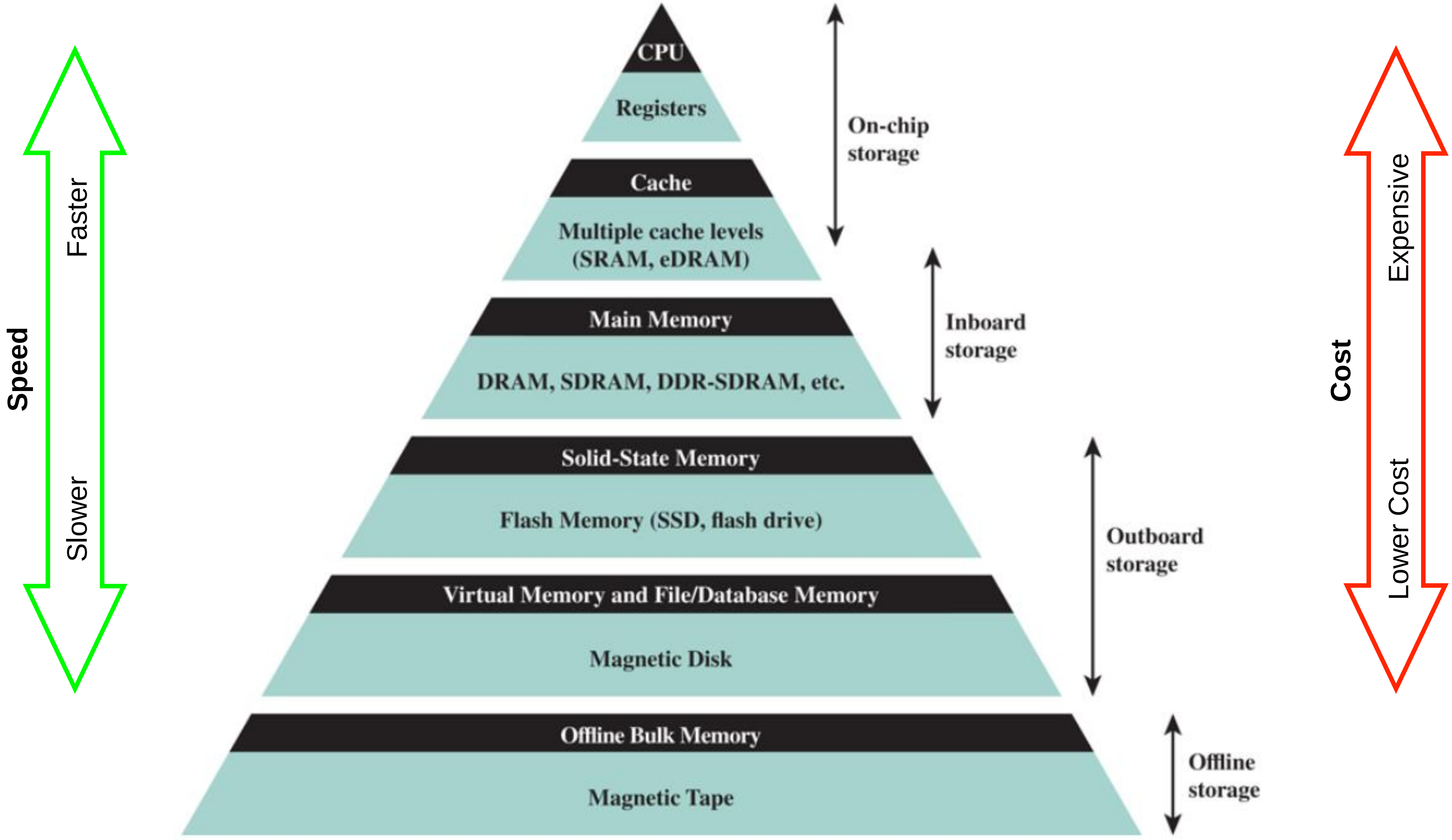
Organization

- Physical arrangement of bits into words
- Not always obvious
- e.g. interleaved organization, characters vs. integers vs. floating point numbers

The Bottom Line: Design Constraints

- **How much?**
  - Capacity
  - Applications tend to be built to use any commonly available amount
- **How fast?**
  - Time is money
  - Must be able to keep up with the CPU
- **How expensive?**
  - Cost must be reasonable

Hierarchy List



Speed

Faster

Slower

CPU

Registers

Cache

Multiple cache levels (SRAM, eDRAM)

Main Memory

DRAM, SDRAM, DDR-SDRAM, etc.

Solid-State Memory

Flash Memory (SSD, flash drive)

Virtual Memory and File/Database Memory

Magnetic Disk

Offline Bulk Memory

Magnetic Tape

On-chip storage

Inboard storage

Outboard storage

Offline storage

Cost

Expensive

Lower Cost

So you want fast?

- It is possible to build a computer which uses only static RAM *(see later)*
- This would be very fast
- This would need no cache
  - How can you cache cache?
- This would cost a very large amount

Locality of Reference

- Success depends upon the **locality of reference**
- During the course of the execution of a program, **memory references tend to cluster** (e.g. loops)
- **Temporal locality**
  - If a location is referenced, it is likely to be referenced again in the near future
- **Positional locality**
  - Time is money
- Basis for caching and virtual memory

Semiconductor Memory

- **RAM**
  - Misnamed as all semiconductor memory is random access
  - Read/Write
  - Volatile
  - Temporary storage
  - Static or Dynamic

Dynamic RAM

- Bits stored as charge in capacitors
- Charges leak
- Need refreshing even when powered
- Simpler construction
- Smaller per bit
- Less expensive
- Need refresh circuits
- Slower
- Main memory

DRAMs in SODIMM Form Factor

Static RAM

- Bits stored as on/off switches
- No charges to leak
- No refreshing needed when powered
- More complex construction
- Larger per bit
- More expensive
- Does not need refresh circuits
- Faster
- Cache

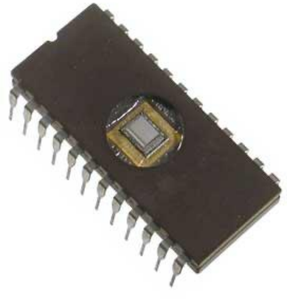
SRAM in an SOIC Package

Read Only Memory (ROM)

- Permanent storage
- Microprogramming *(see later)*
- Library subroutines
- Systems programs (BIOS)
- Function tables

Types of ROM

- Written during manufacture
  - Very expensive for small runs
- **Programmable (once) - PROM**
  - Needs special equipment to program
- **Read "mostly"**
  - **Erasable Programmable (EPROM)**
    - Erased by UV
  - **Electrically Erasable (EEPROM)**
    - Byte-level writing
    - Takes much longer to write than read
  - **Flash memory**
    - Erase whole memory electrically



EPROM in DIP Package

Flash memory in Quad-Flat Package

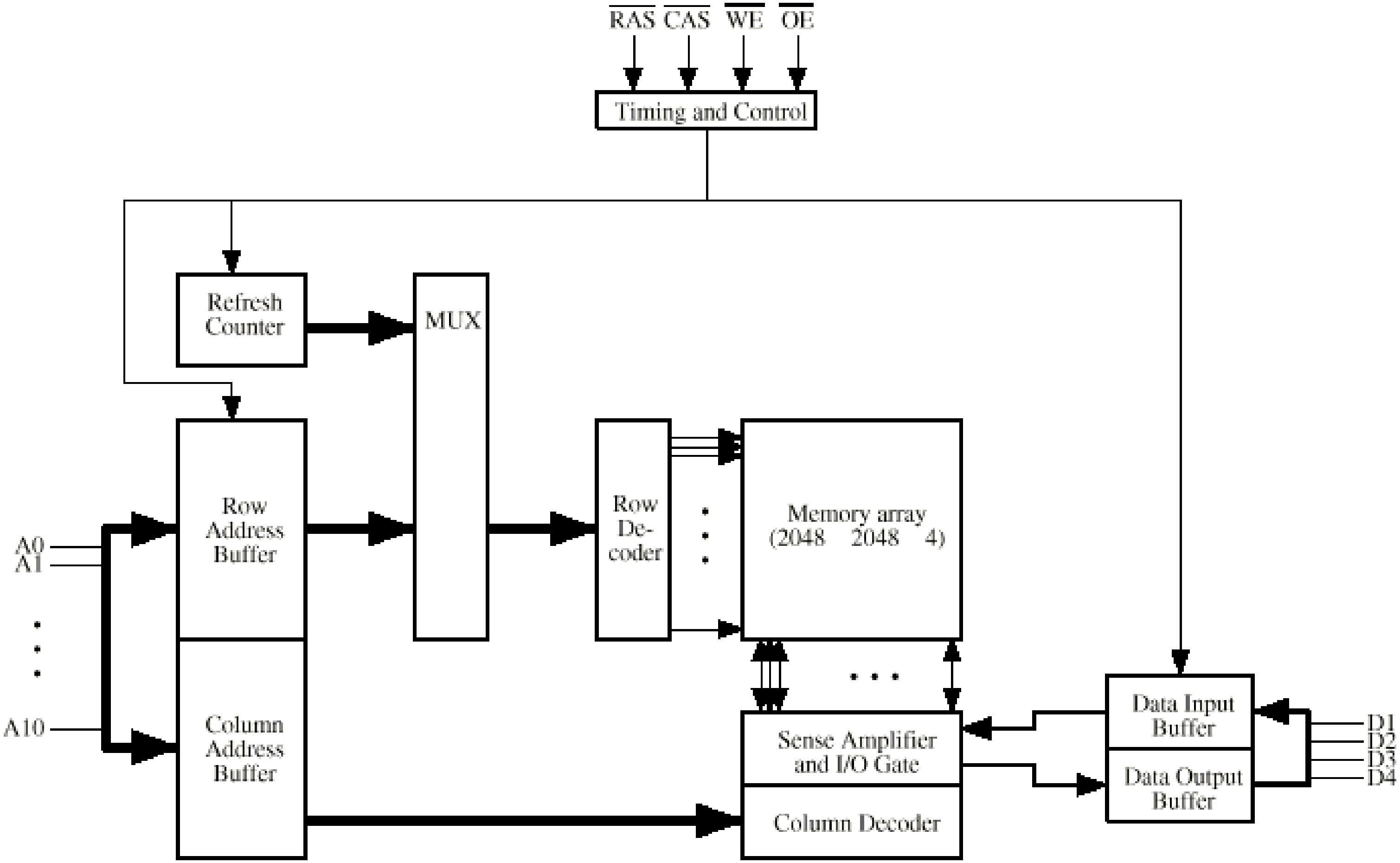
Organization in detail

- A 16Mbit chip can be organized as 1M of 16 bit words
- A bit per chip system has 16 lots of 1Mbit chip with bit 1 of each word in chip 1 and so on
- A 16Mbit chip can be organized as a 2048 x 2048 x 4bit array
  - Reduces number of address pins
    - Multiplex row address and column address
    - 11 pins to address (2^{11}=2048)
    - Adding one more pin doubles range of values

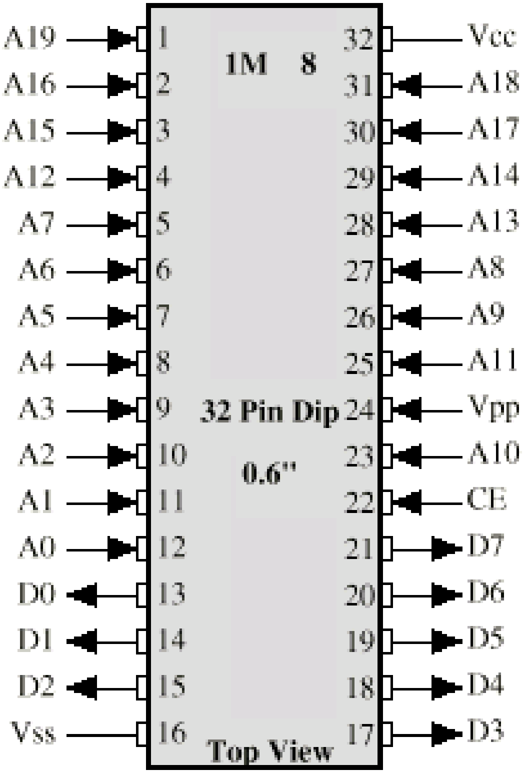
Refreshing

- Refresh circuit included on chip
- Disable chip
- Count through rows
- Read & Write back
- Takes time
- Slows down apparent performance

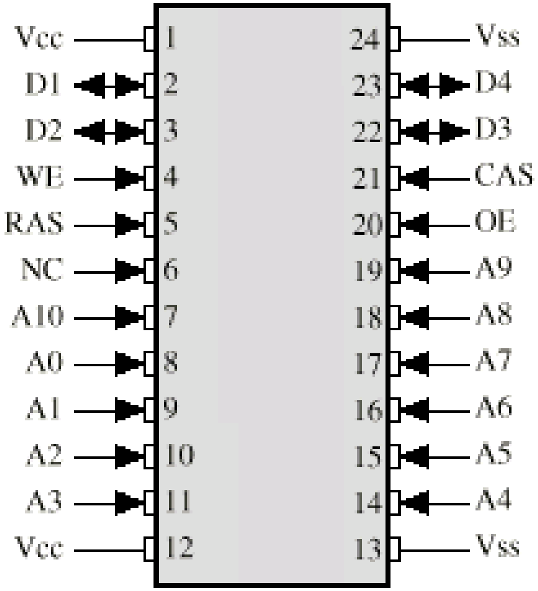
Typical 16 Mb DRAM (4M x 4)



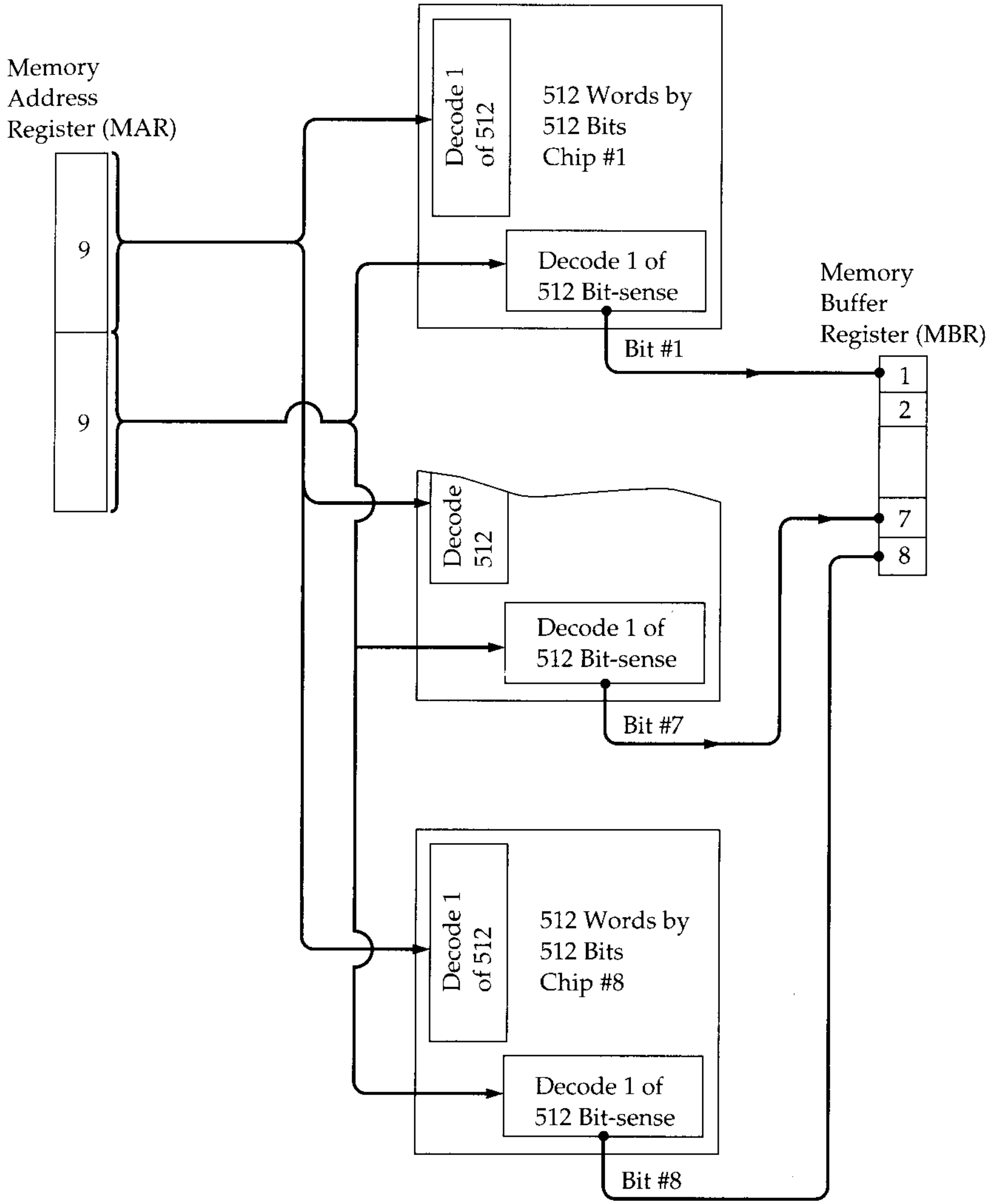
Packaging



(a) 8 Mbit EPROM

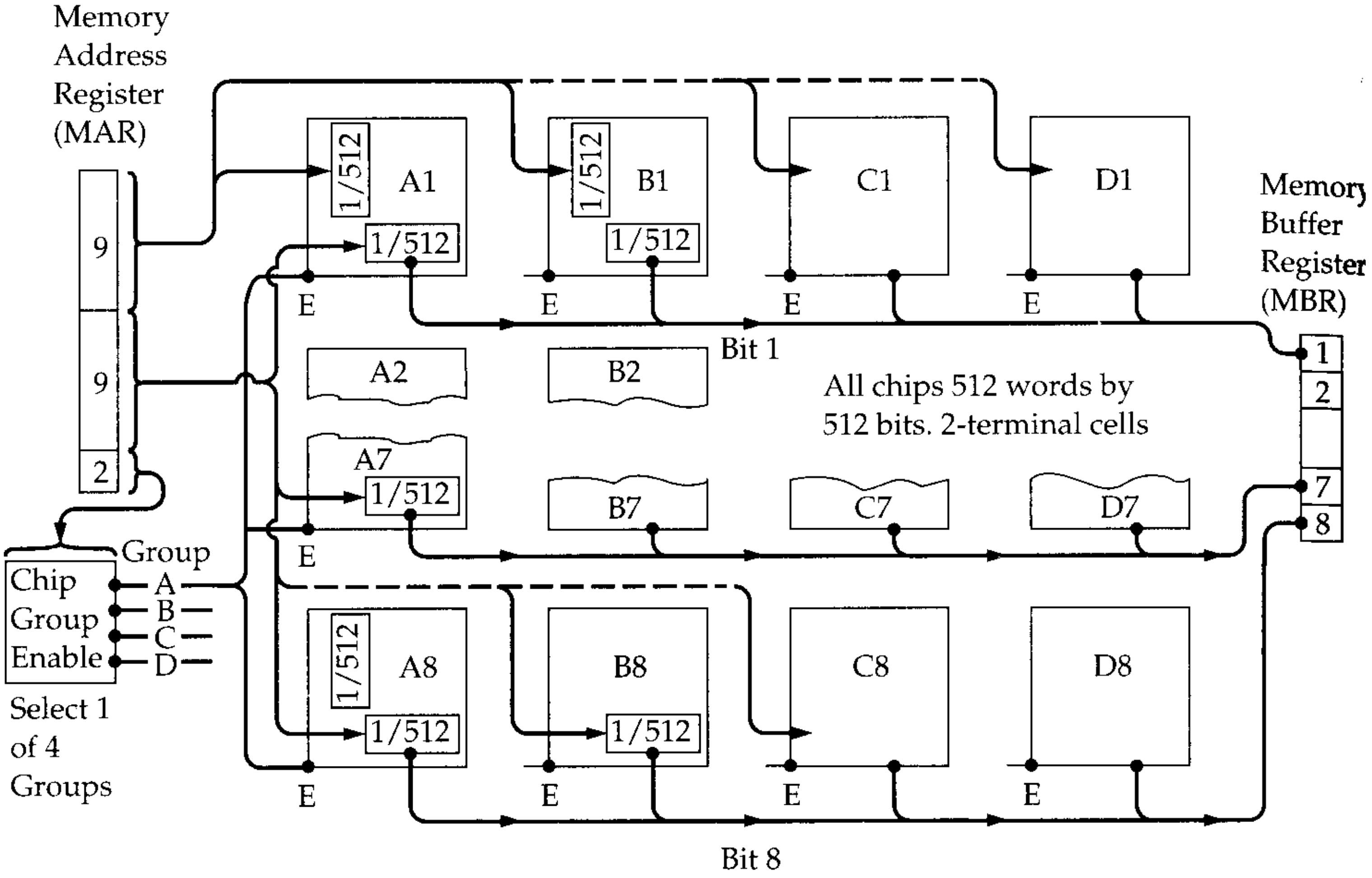


(b) 16 Mbit DRAM



Module organization

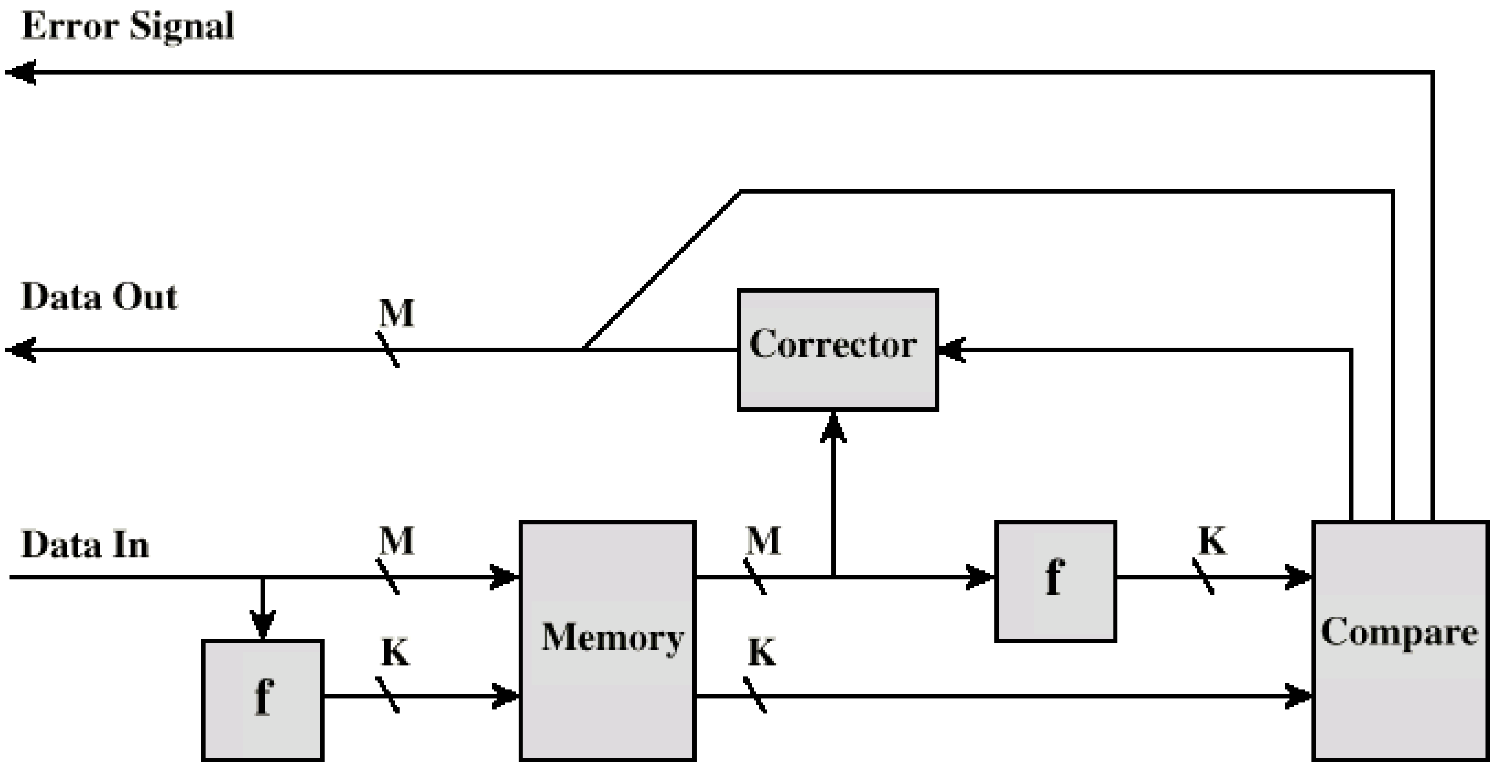
Module organization (2)



Error Correction

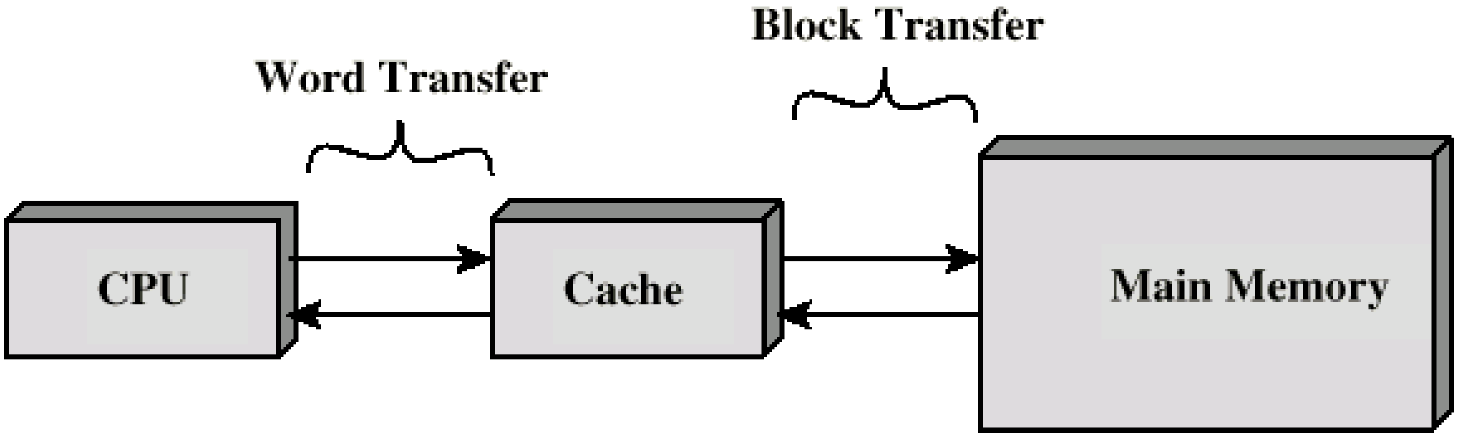
- **Hard Failure**
  - Permanent defect
- **Soft Error**
  - Random, non-destructive
  - No permanent damage to memory
  - Caused by power supply problems, alpha particles
- **Detection and Correction**
  - Hamming error correcting code
  - SEC-DED codes

Error Correcting Code Function



Cache

- Small amount of fast memory
- Sits between normal main memory and CPU
- May be located on CPU chip or module



Cache operation - overview

- CPU requests contents of memory location
- Check cache for this data
- If present, get from cache (fast)
- If not present, read required block from main memory to cache
- Then deliver from cache to CPU
- Cache includes **tags** to identify which block of main memory is in each cache slot
- The proportion of memory references, which are found already stored in cache, is called the **hit ratio**.

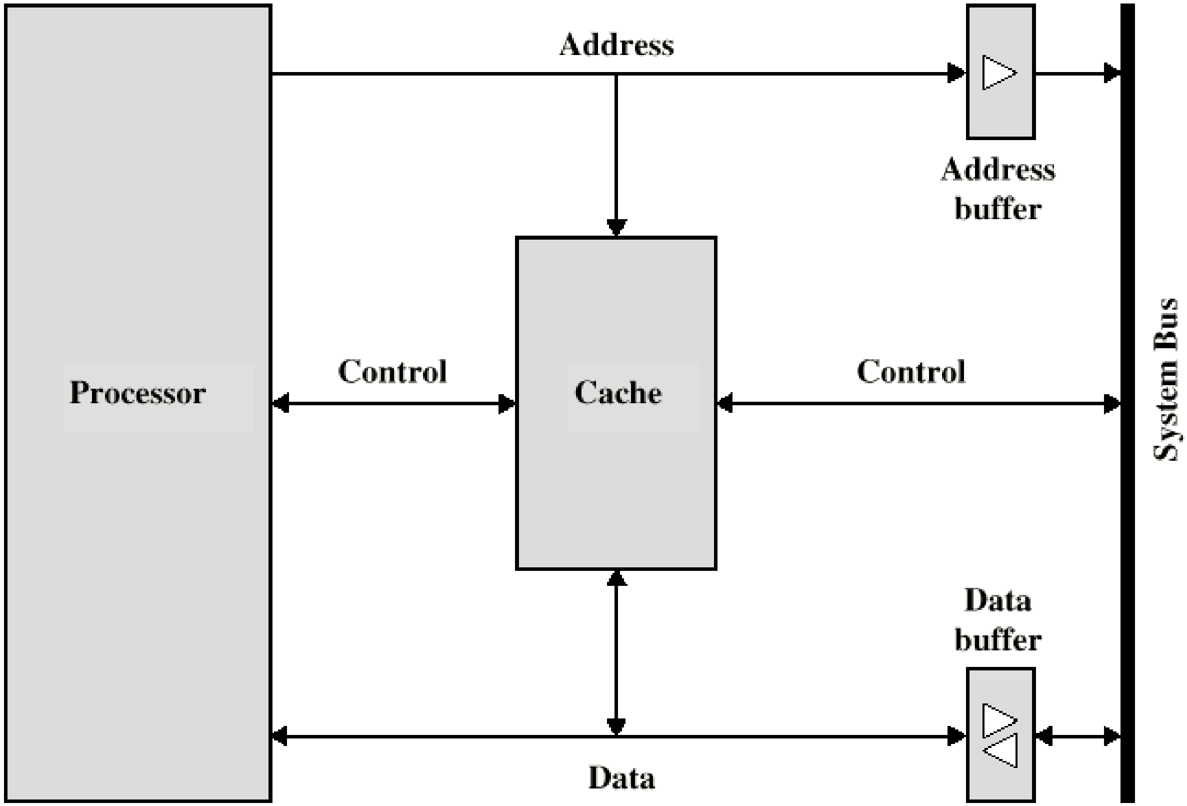
Elements of Cache Design

- Size
- Mapping Function
- Replacement Algorithm
- Write Policy
- Block Size
- Number of Caches

Size does matter

- **Cost**
  - More cache is expensive
- **Speed**
  - More cache is faster (up to a point)
  - Checking cache for data takes time

Typical Cache Organization



Mapping Function

- Determines which cache line to use for a particular block of main memory *(since number_of_cache_lines* << *number_of_blocks)*
- Cache of 64kByte
- Cache block of 4 bytes
  - i.e. cache is 16k (2^{14}) lines of 4 bytes
- 16MBytes main memory
- 24 bit address
  - (2^{24}=16M)

Direct Mapping

- Each block of main memory maps to only one cache line
  - i.e. if a block is in cache, it must be in one specific place
- **Address** is in two parts
- Least Significant *w* bits identify **unique word**
- Most Significant *s* bits specify **one memory block**
- The MSBs are split into a cache line field *r* and a tag of *s*-*r* (most significant)

Direct Mapping Address Structure

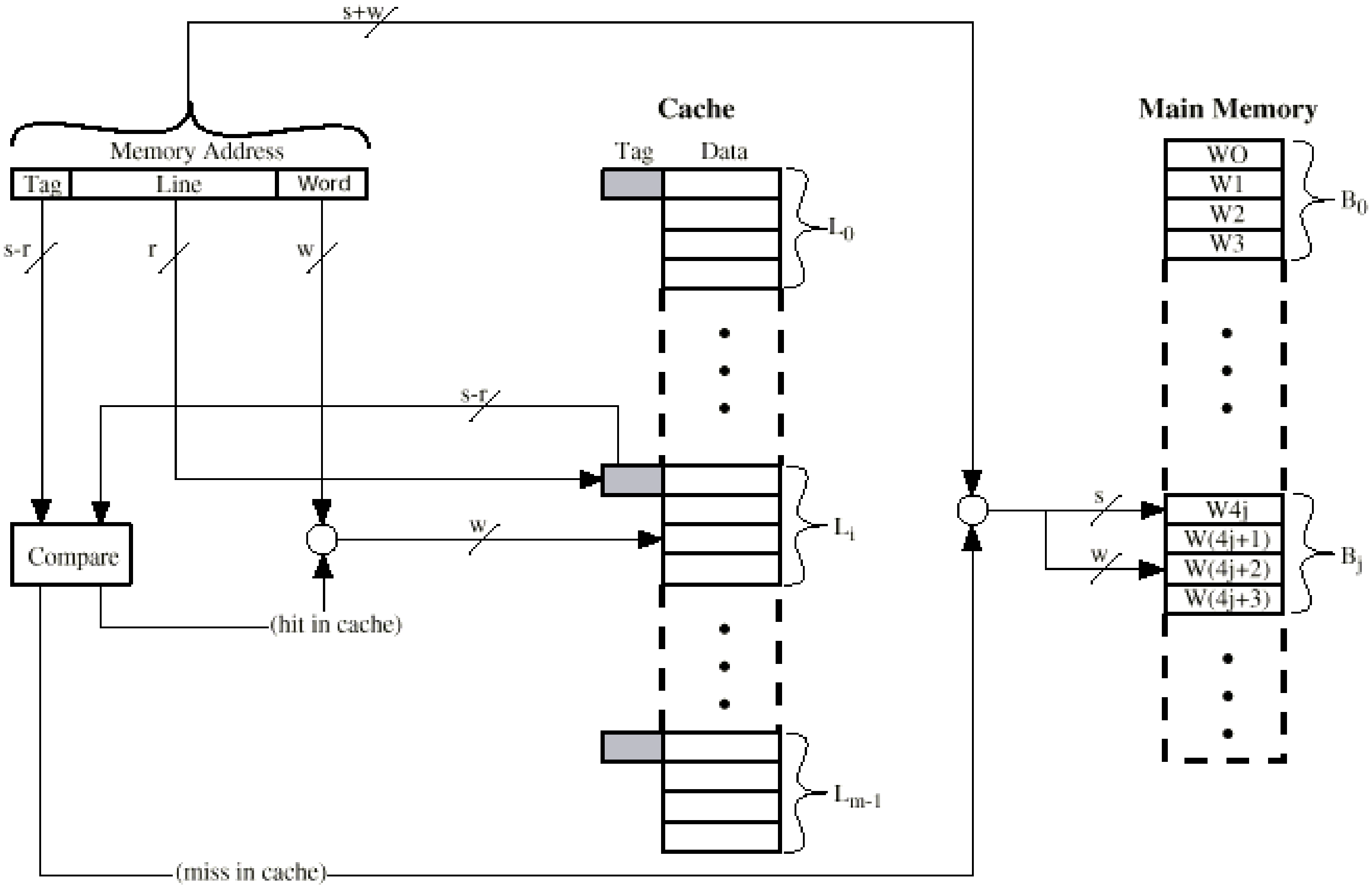
Tag s-r	Line or Slot r	Word w
8	14	2

- 24 bit address
- 2 bit word identifier (4 byte block)
- 22 bit block identifier
  - – 8 bit tag (=22-14)
  - – 14 bit slot or line
- No two blocks in the same line have the same Tag field
- Check contents of cache by finding line and checking Tag

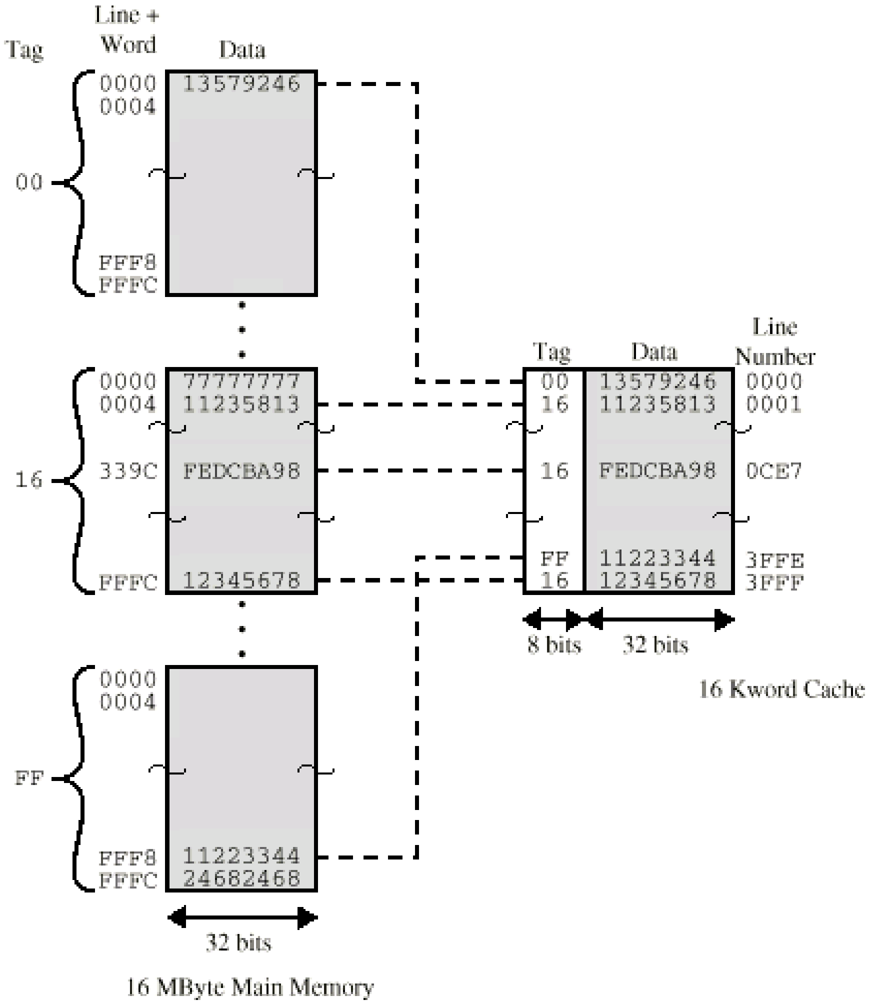
Direct Mapping Cache Line Table

Cache line	Main Memory blocks held
0	0, m, 2m, 3m…2^s-m
1	1,m+1, 2m+1…2^s-m+1
m-1	m-1, 2m-1,3m-1…2^s-1

Direct Mapping Cache Organization



Direct Mapping Example



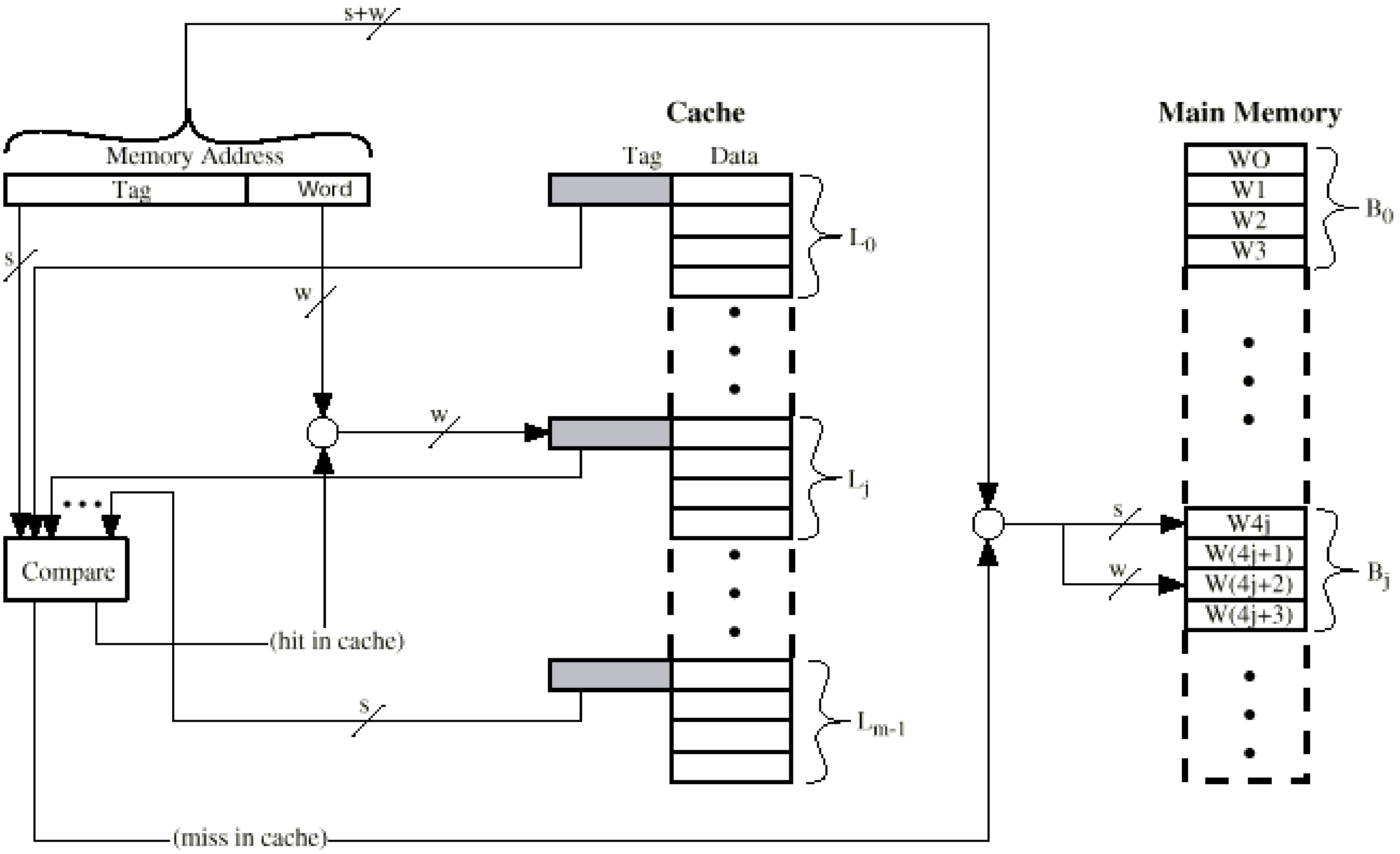
Direct Mapping pros & cons

- Simple
- Inexpensive
- Fixed location for given block
  - If a program accesses 2 blocks that map to the same line repeatedly, cache misses are very high

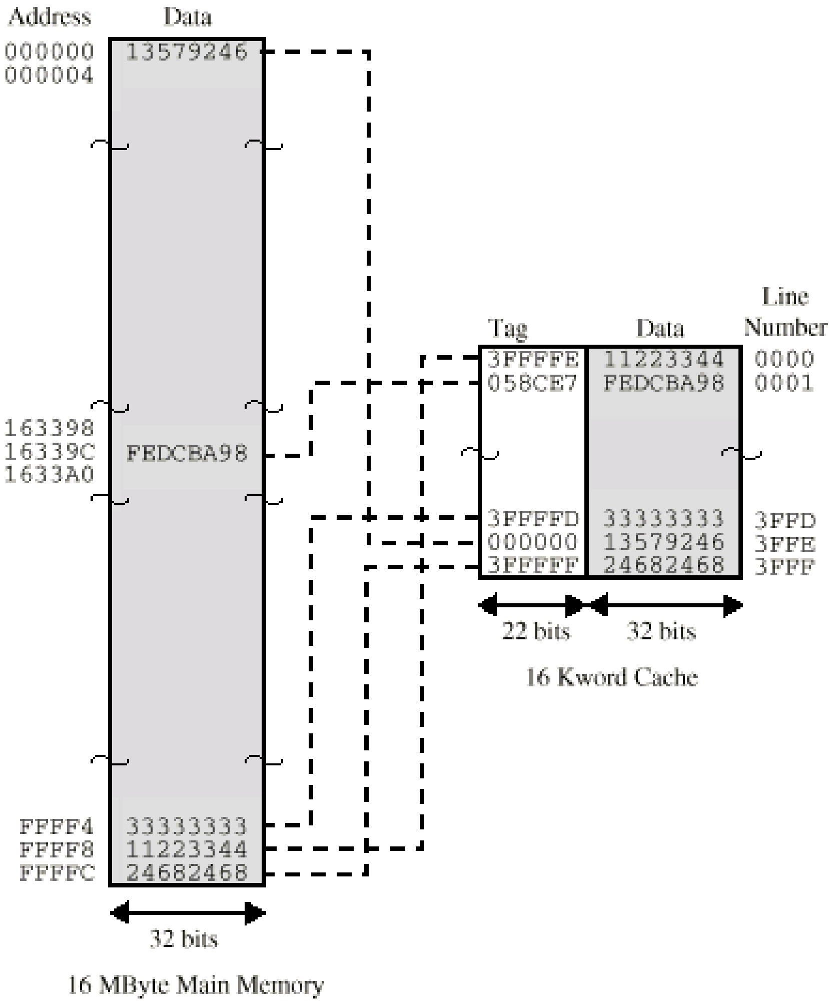
Associative Mapping

- A main memory block can load into any line of cache
- Memory address is interpreted as tag and word
- Tag uniquely identifies block of memory
- Every line's tag is examined for a match
- Cache searching gets expensive

Fully Associative Cache Organization



Associative Mapping Example



Address	Data
000000	13579246
000004	
163398	
16339C	FEDCBA98
1633A0	
FFFF4	33333333
FFFF8	11223344
FFFFC	24682468

Tag	Data	Line Number
3FFFFE	11223344	0000
058CE7	FEDCBA98	0001
3FFFFD	33333333	3FFD
000000	13579246	3FFE
3FFFFF	24682468	3FFF

Associative Mapping Address Structure

Tag 22 bit	Word 2 bit

- 22 bit tag stored with each 32 bit block of data
- Compare tag field with tag entry in cache to check for hit
- Least significant 2 bits of address identify which 16 bit word is required from 32 bit data block
- e.g.

Address	Tag	Data	Cache line
FFFFFC	FFFFFC	24682468	3FFF

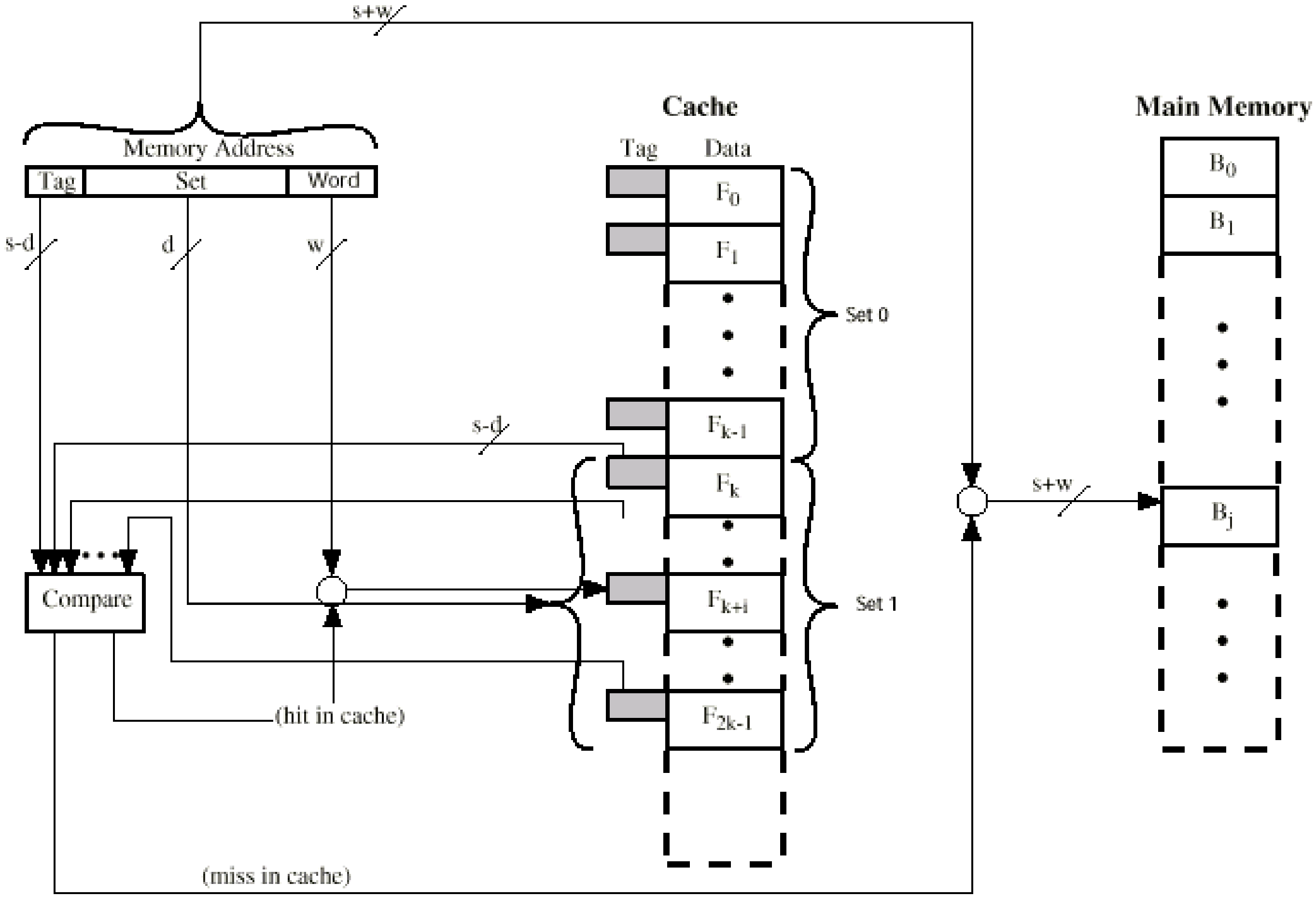
Set Associative Mapping

- Cache is divided into a number of sets
- Each set contains a number of lines
- A given block maps to any line in a given set
  - e.g. Block B can be in any line of set i
- e.g. 2 lines per set
  - 2 way associative mapping
  - A given block can be in one of 2 lines in only one set

Set Associative Mapping Example

- 13 bit set number
- Block number in main memory is modulo 2^{13}
- 000000, 00A000, 00B000, 00C000 … map to same set

Two Way Set Associative Cache Organization

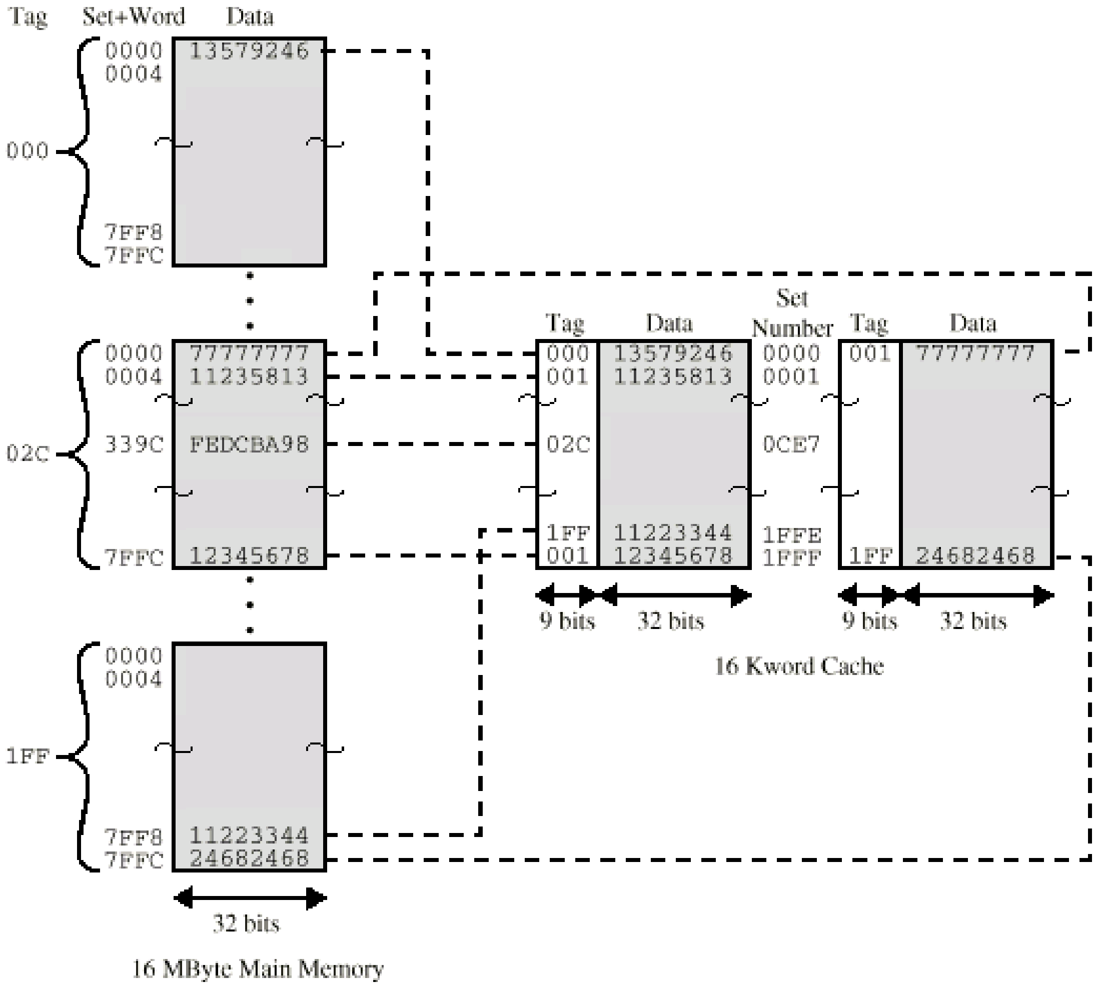


Set Associative Mapping Address Structure

Tag 9 bit	Set 13 bit	Word 2 bit

- Use set field to determine cache set to look in
- Compare tag field to see if we have a hit
- e.g
  - Address Tag Data Set number
  - 1FF 7FFC 1FF 123456781FFF
  - 001 7FFC 001 112233441FFF

Two Way Set Associative Mapping Example



Replacement Algorithms (1)
Direct mapping

- No choice
- Each block only maps to one line
- Replace that line

Replacement Algorithms (2)
Associative & Set Associative

- Hardware implemented algorithms (speed)
- **Least Recently used (LRU)**
- e.g. in 2 way set associative
  - Which of the 2 block is LRU?
- **First in first out (FIFO)**
  - replace block that has been in cache longest
- **Least frequently used**
  - replace block which has had fewest hits
- **Random**

Write Policy

- Must not overwrite a cache block unless main memory is up to date
- Multiple CPUs may have individual caches
- I/O may address main memory directly

Write through

- All writes go to main memory as well as cache
- Multiple CPUs can monitor main memory traffic to keep local (to CPU) cache up to date
- Lots of traffic
- Slows down writes
- Generates substantial memory traffic and may create a bottleneck

Write back

- Updates initially made in cache only
- Update bit for cache slot is set when update occurs
- If block is to be replaced, write to main memory only if update bit is set
- Other caches get out of sync
- I/O must access main memory through cache
- N.B. 15% of memory references are writes

Pentium Cache

- Foreground reading
- Find out detail of Pentium II cache systems
- NOT just from Stallings!

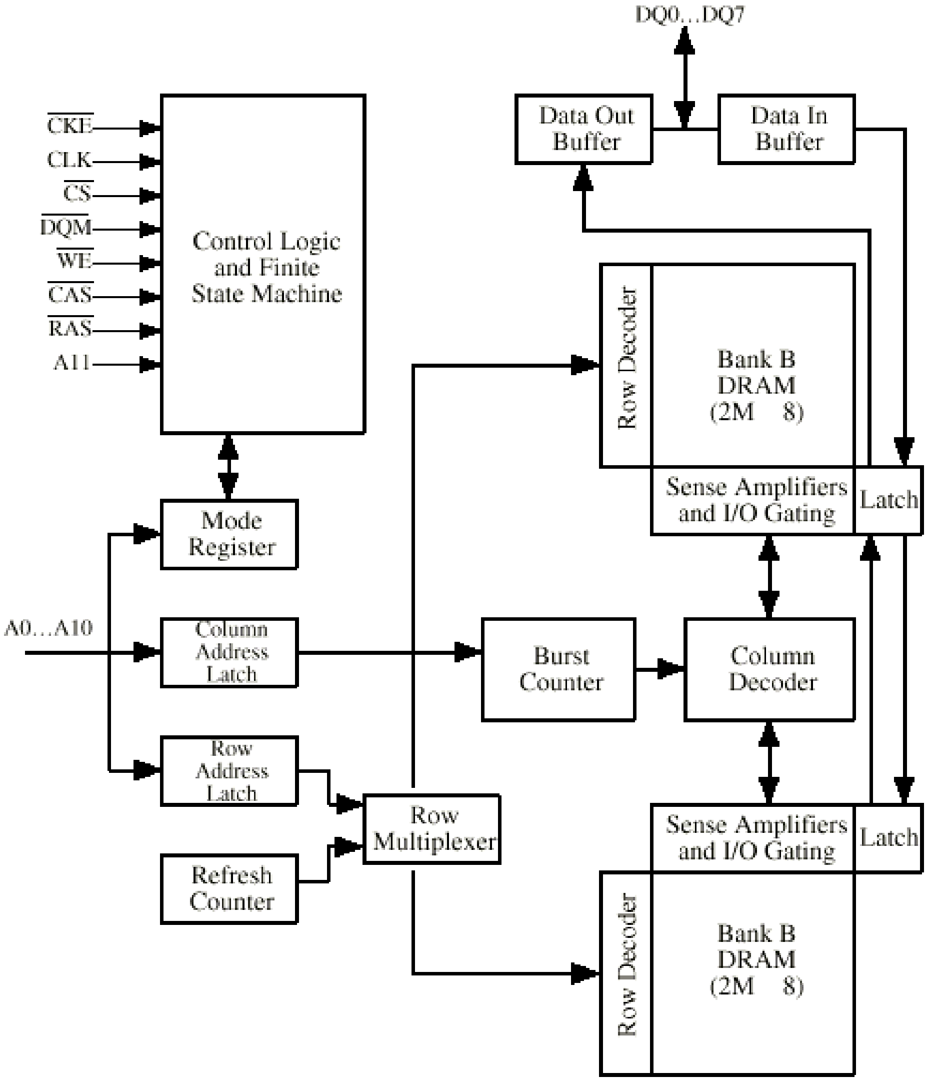
New RAM Technology (1)

- **Basic DRAM** same since first RAM chips
- **Enhanced DRAM**
  - Contains small SRAM as well
  - SRAM holds last line read (c.f. Cache!)
- **Cache DRAM**
  - Larger SRAM component
  - Use as cache or serial buffer

New RAM Technology (2)

- **Synchronous DRAM (SDRAM)**
  - currently on DIMMs
  - Access is synchronized with an external clock
  - Address is presented to RAM
  - RAM finds data (CPU waits in conventional DRAM)
  - Since SDRAM moves data in time with system clock, CPU knows when data will be ready
  - CPU does not have to wait, it can do something else
  - Burst mode allows SDRAM to set up stream of data and fire it out in block

SDRAM

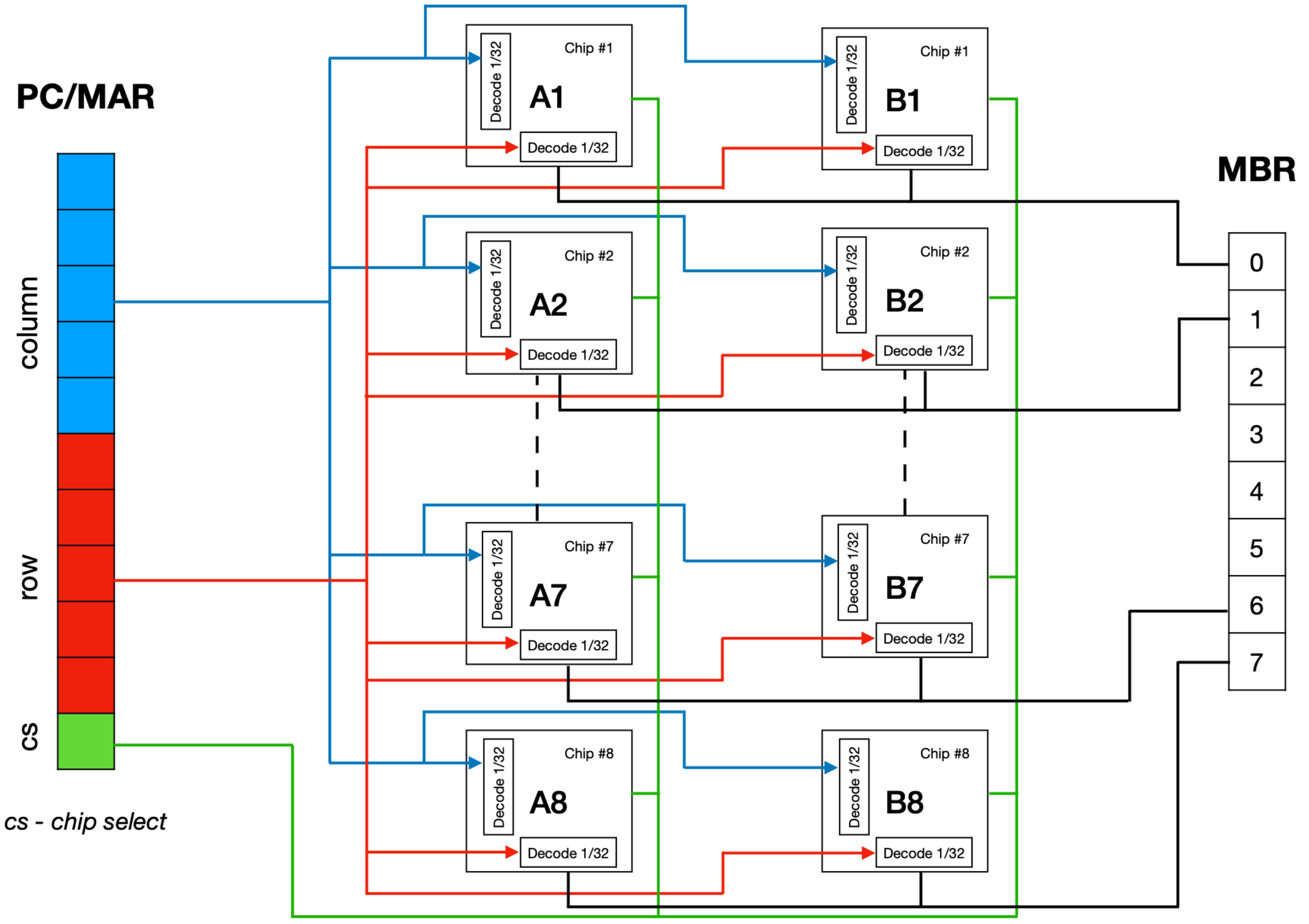

DQ0…DQ7
Data Out Buffer
Data In Buffer
$\overline{\text{CKE}}$
CLK
$\overline{\text{CS}}$
$\overline{\text{DQM}}$
$\overline{\text{WE}}$
$\overline{\text{CAS}}$
$\overline{\text{RAS}}$
A11
Control Logic and Finite State Machine
Row Decoder
Bank B DRAM (2M 8)
Sense Amplifiers and I/O Gating
Latch
Mode Register
A0…A10
Column Address Latch
Burst Counter
Column Decoder
Row Address Latch
Row Multiplexer
Refresh Counter
Sense Amplifiers and I/O Gating
Latch
Row Decoder
Bank B DRAM (2M 8)

New RAM Technology (3)

- Foreground reading
- Check out any other RAM you can find
- See Web site:
  - The RAM Guide

TRACS Memory Organization (RAM)

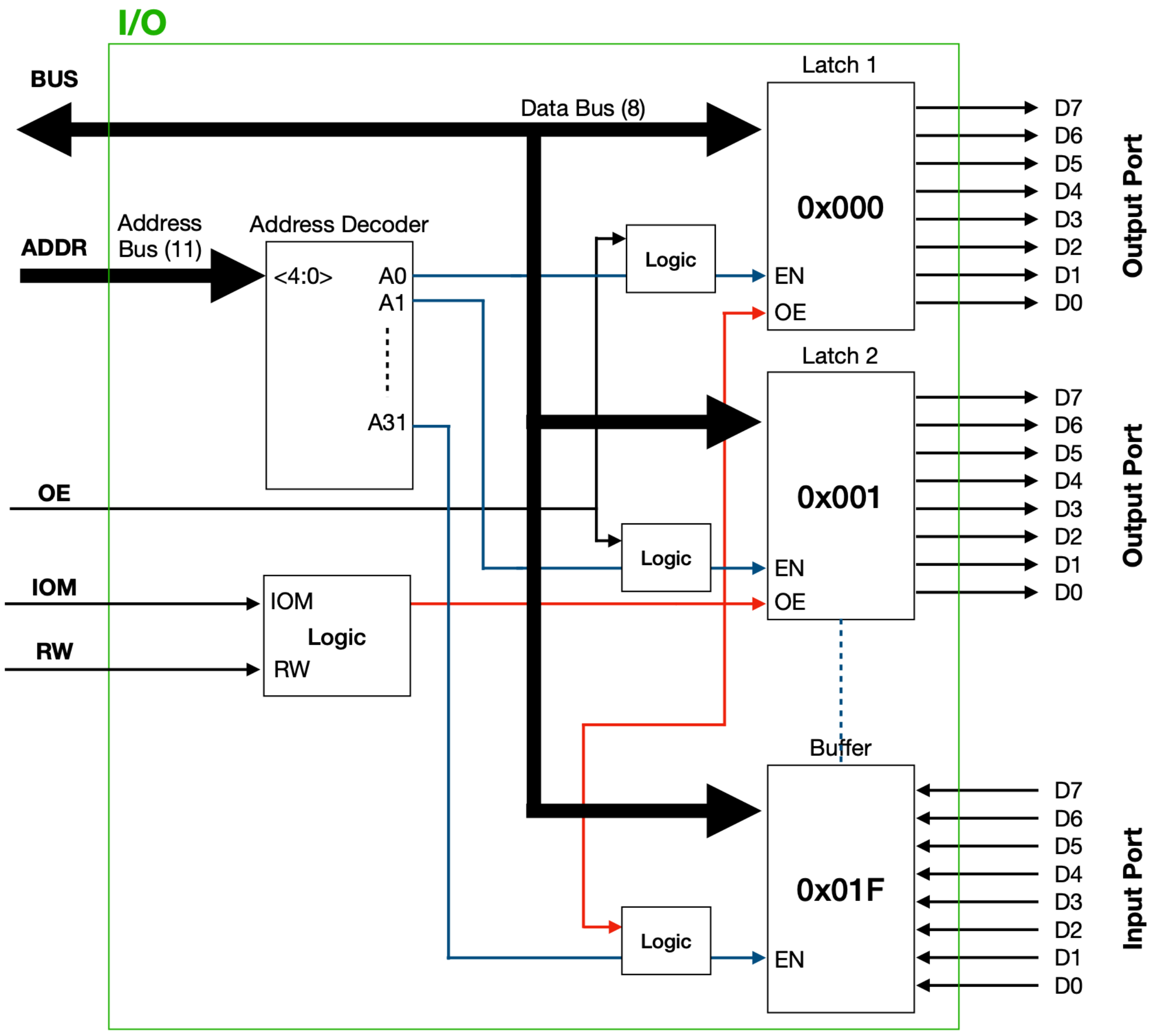
- Composed of **sixteen (16) 32x32 bit chips** for total of **16Kbit**
- **Two (2) chip groups** A & B
- Organized into (32 + 32) bit columns and (32 x 8) rows
- **Chip Select (cs)** identify the chip group A or B
- **Address bus** is composed of *column address*, *row address* and *chip group selector*



TRACS Memory Organization

TRACS Memory Organization (I/O)

- Isolated from Main Memory (as I/O Memory)
- **Maximum of 32 addressable latch/buffer**
- **IO address decoding**, enables the proper latch or buffer
- **Latches for output signals** and **buffer for input signals**
- **I/O read and write control** via IOM and RW control lines
- **Data latched (output)** when IOM=0 and RW=1
- **Data buffered (input)** when IOM=0 and RW=0



TRACS I/O Memory Organization

University of San Carlos | Department of
COMPUTER ENGINEERING

CpE 3202
Computer Organization & Architecture

End of Lecture

Note: The lecture slide was based on the accompanying lecture material from the Computer Organization and Architecture textbook by William Stallings. Some contents are contributed by Van B. Patiluna (USC). All contents contributed are copyright to the respective authors. For instructional use only. **Do not distribute.**

References:

- Brey, Barry B. *The Intel microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro processor, Pentium II, Pentium III, Pentium 4, and Core2 with 64-bit extensions: architecture, programming, and interfacing* / Barry B. Brey—8th ed.
- Stallings, W. Computer Organization and Architecture, 6th edition, Pearson Education, Inc. (2003).