COSC4201

Chapter 5
Main Memory

Prof. Mokhtar Aboelaze

York University

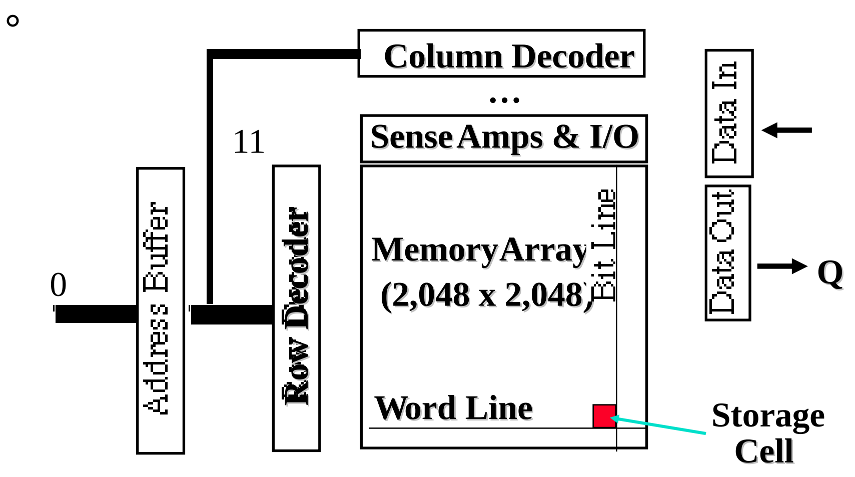
Main Memory

- Main memory generally utilizes Dynamic RAM (DRAM), which use a single transistor to store a bit, but require a periodic data refresh by reading every row (~every 8 msec).
- Static RAM may be used for main memory if the added expense, low density, high power consumption, and complexity is feasible (e.g. Cray Vector Supercomputers).
- Main memory performance is affected by:
  - Memory latency: Affects cache miss penalty. Measured by:
    - **Access time: The time it takes between a memory access request is issued to main memory and the time the requested information is available to cache/CPU.**
    - **Cycle time: The minimum time between requests to memory (greater than access time in DRAM to allow address lines to be stable)**
  - Memory bandwidth: The maximum sustained data transfer rate between main memory and cache/CPU.

Main memory

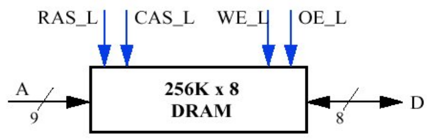
- t_{RAC}: Minimum time from RAS (Row Access Strobe) line falling to the valid data output.
  - Usually quoted as the nominal speed of a DRAM chip
  - For a typical 4Mb DRAM t_{RAC} = 60 ns
- t_{RC}: Minimum time from the start of one row access to the start of the next.
  - t_{RC} = 110 ns for a 4Mbit DRAM with a t_{RAC} of 60 ns
- t_{CAC}: minimum time from CAS (Column Access Strobe) line falling to valid data output.
  - 15 ns for a 4Mbit DRAM with a t_{RAC} of 60 ns
- t_{PC}: minimum time from the start of one column access to the start of the next.
  - About 35 ns for a 4Mbit DRAM with a t_{RAC} of 60 ns

Logical DRAM Organization



- Square root of N (size) per RAS/CAS

DRAM



- Control Signals (RAS_L, CAS_L, WE_L, OE_L) are all active low
- Din and Dout are combined (D):
  - WE_L is asserted (Low), OE_L is disasserted (High)
    - D serves as the data input pin
  - WE_L is disasserted (High), OE_L is asserted (Low)
    - D is the data output pin
- Row and column addresses share the same pins (A)
  - RAS_L goes low: Pins A are latched in as row address
  - CAS_L goes low: Pins A are latched in as column address

DRAM

- Regular DRAM Organization
  - N rows x N columns x M bits
  - Read/Write M bits at a a time (RAS + CAS)
- Fast Page DRAM
  - Need NM registers (SRAM) to store the row
  - Then we need CAS only to read each word
- Extended Data Out (EDO) DRAM
  - Extended Data Out DRAM operates in a similar fashion to Fast Page Mode DRAM except the data from one read is on the output pins at the same time the column address for the next read is being latched in.

Improving main memory Performance

° **Higher Bandwidth**

- By increasing the bus width, we decrease the miss penalty.
- Expensive
- Need a multiplexer since the CPU usually access one word only (CPU cache bus width should be 1 word), MUX is on the critical path

Improving main memory Performance

° **Simple Interleaved memory**

- Memory is organized in banks
- Send the address to the 4 banks in the same time (or interleaved)
- Data are read from one bank at a time.

° **Independent Memory banks**

- Each bank has a separate address line

Example

Given the following system parameters with single cache level L_1:

Block size=1 word Memory bus width=1 word Miss rate =3% Miss penalty=32 cycles
(4 cycles to send address 24 cycles access time/word, 4 cycles to send a word)
Memory access/instruction = 1.2 Ideal CPI (ignoring cache misses) = 2
Miss rate (block size=2 word)=2% Miss rate (block size=4 words) =1%

- **The CPI of the base machine with 1-word blocks = 2+(1.2 x 0.03 x 32) = 3.15**
- **Increasing the block size to two words gives the following CPI:**
  - 32-bit bus and memory, no interleaving = 2 + (1.2 x .02 x 2 x 32) = 3.54
  - 32-bit bus and memory, interleaved = 2 + (1.2 x .02 x (4 + 24 + 8) = 2.86
  - 64-bit bus and memory, no interleaving = 2 + (1.2 x 0.02 x 1 x 32) = 2.77
- **Increasing the block size to four words; resulting CPI:**
  - 32-bit bus and memory, no interleaving = 2 + (1.2 x 0.01 x 4 x 32) = 3.54
  - 32-bit bus and memory, interleaved = 2 + (1.2 x 0.01 x (4 +24 + 16) = 2.53
  - 64-bit bus and memory, no interleaving = 2 + (1.2 x 0.01 x 2 x 32) = 2.77

Avoiding Memory Bank Conflicts

- **Suppose that we have 128 banks, and we will store 512x512 array.**
- **All the elements of a row will be mapped to the same bank (conflicts if we access a row.**
- **Usually, the number of banks is a power of 2, in this case**
- **Bank number = address MOD number of banks**
- **Address within a bank =Address/Number of banks**
- **This is a trivial calculation if the number of banks is a power of 2.**
- **If the number of memory banks is a prime number, that will decrease conflicts, but division and MOD will be very expensive**

Avoiding Memory Banks Conflicts

- MOD can be calculated very efficiently if the prime number is 1 less than a power of 2.
- Division still a problem
- But if we change the mapping such that
- Address in a bank = address MOD number of words in a bank.
- Since the number of words in a bank is usually a power of 2, that will lead to a very efficient implementation.
- Consider the following example, the first case is the usual 4 banks, then 3 banks with sequential interleaving and modulo interleaving and notice the conflict free access to rows and columns of a 4 by 4 matrix

Example

Add in a bank					SE	Q		M	O	D
	0	1	2	3	0	1	2	0	1	2
0	0	1	2	3	0	1	2	0	16	8
1	4	5	6	7	3	4	5	9	1	17
2	8	9	10	11	6	7	8	18	10	2
3	12	13	14	15	9	10	11	3	19	11
4	16	17	18	19	12	13	14	12	4	20
5	20	21	22	23	15	16	17	21	13	5
6	24	25	26	27	18	19	20	6	22	14
7	28	29	30	31	21	22	23	15	7	23