Lecture No 15

Error Detection & Detection

Error Detection and Correction

- DRAM cells using very high density have very small size.
- Each cell thus carries very small amount of charge to determine data state.
- Chances of corruptions are very high due to environmental perturbations, static electricity etc.
- Error detection and correction is thus intrinsic part of memory system design.

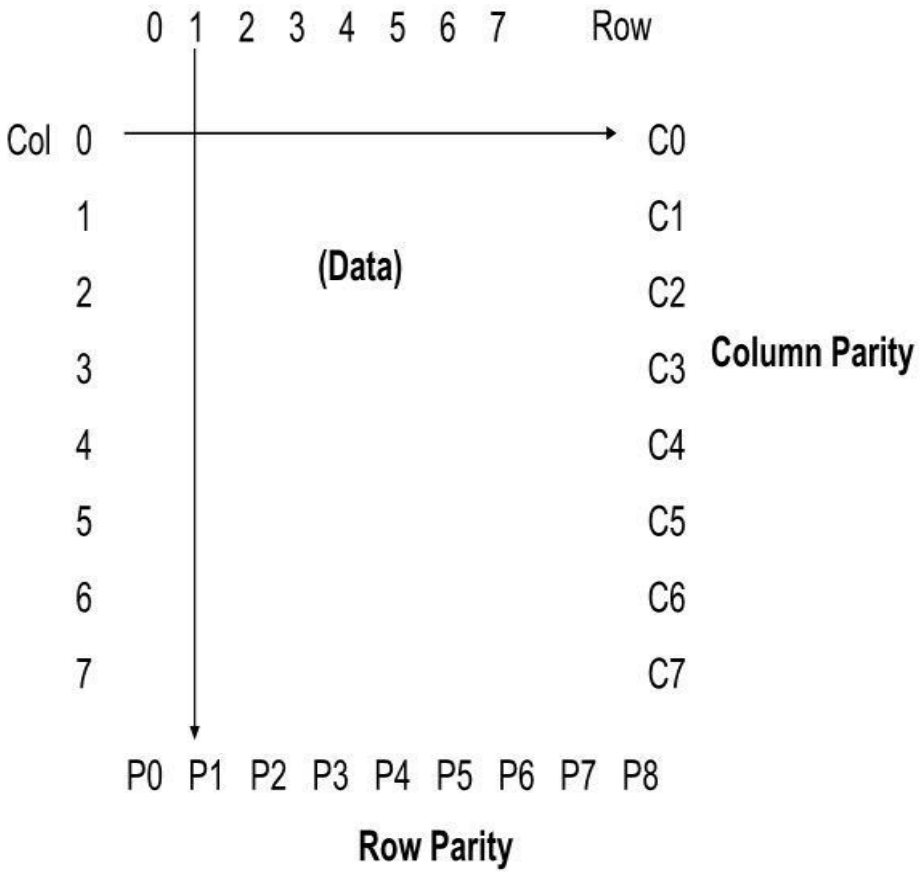
Error Detection and Correction

- Simplest type of error detection is Parity.
- A bit called parity bit is added to each memory word, which ensures that the sum of the number of 1's in the word is even (or odd).
- If a single error occurs to any bit in the word, the sum modulo 2 of the number of 1's in the word is inconsistent with parity assumption and word is known to have been corrupted.

Error Detection and Correction

- Most modern memories incorporate hardware to automatically correct single errors (ECC – error correcting codes)
- The simplest code of this type might consist of a geometric block code
- The message bits to be checked are arranged in a roughly square pattern and each column and row is augmented with a parity bit.
- If a row and column indicate a flaw when decoded at receiver end, then fault lies at the intersection bit which can be simply inverted for error correction.

Two Dimensional ECC



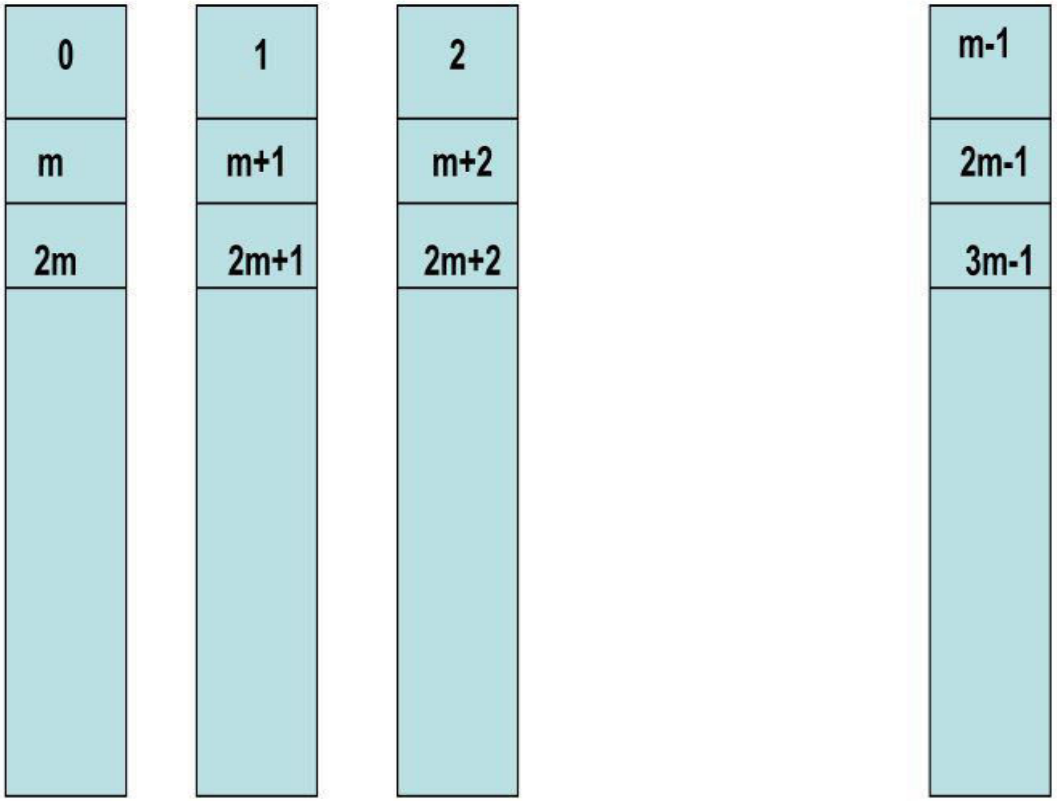
Error Detection and Correction

- For 64 message bits we need to add 17 parity bits, 8 for each of the rows and column and one additional parity bit to compute parity on the parity row and column.
- If failure is noted in a single row or a single column or multiple rows and columns then it is a case of multi bit failure and a non correctable state is entered.

Achieved Memory Bandwidth

- Two factors have substantial effect on achieved memory bandwidth.
  - **Memory Buffers :** Buffering should be provided for memory requests in the processor or memory system until the memory reference is complete. This maximizes requests made by the processor resulting in possible increase in achieved bandwidth.
  - **Partitioning of Address Space:** The memory space should be partitioned in such a manner that memory references are equally distributed across memory modules.

Assignment of Address Space to m Memory Modules



Interleaved Memory System

- Partitioning memory space in m memory modules is based on the premise that successive references tend to be successive memory locations.
- Successive memory locations are assigned to distinct memory modules.
- For m memory modules an address x is assigned to a module x mod m.
- This partitioning strategy is termed interleaved memory system and no of modules m is the degree of interleaving.

Interleaved Memory System

- Since m is a power of two so x mod m results in memory module to be referenced, being determined by low order bits of the memory address.
- This is called ***low order interleaving***.
- Memory addresses can also be mapped to memory modules by ***higher order interleaving***
- In higher order interleaving upper bits of memory address define a module and lower bits define a word in that module

Interleaved Memory System

- In higher order interleaving most of the references tend to remain in a particular module whereas in low order interleaving the references tend to be distributed across all the modules.
- Thus low order interleaving provides for better memory bandwidth whereas higher order interleaving can be used to increase the reliability of memory system by reconfiguring memory system.

Memory Systems Design

- High performance memory system design is an iterative process.
- Bandwidth and partitioning of the system are determined by evaluation of cost , access time and queuing requirements.
- More modules provide more interleaving and more bandwidth, reduce queuing delay and improve access time.
- But it increases system cost and interconnect network becomes more complex, expensive and slower.

Memory Systems Design

The Basic design steps are as follows:

1. **Determine number of memory modules** and the partitioning of memory system.
2. **Determine offered bandwidth.:** Peak instruction processing rate multiplied by expected memory references per instruction multiplied by number of processors.
3. **Decide interconnection network:** Physical delay through the network plus delays due to network contention cause reduced bandwidth and increased access time. High performance time multiplexed bus or crossbar switch can reduce contention but increases cost.

Memory Systems Design

4. **Assess Referencing Behavior:** Program behavior in its sequence of requests to memory can be

   - Purely sequential: each request follows a sequence.

   - Random: requests uniformly distributed across modules.

   -Regular: Each access separated by a fixed number (Vector or array references)

Random request pattern is commonly used in memory systems evaluation.

Memory Systems Design

5. **Evaluate memory model:** Assessment of ***Achieved Bandwidth*** and ***actual memory access time*** and the ***queuing*** required in the memory system in order to support the achieved bandwidth.

Memory Models

Nature of Processor:

- **Simple Processor:** Makes a single request and waits for response from memory.
- **Pipelined Processor:** Makes multiple requests for various buffers in each memory cycle
- **Multiple Processors:** Each requesting once every memory cycle.

Single processor with n requests per memory cycle is asymptotically equivalent to n processors each requesting once every memory cycle.

Memory Models

Achieved Bandwidth: Bandwidth available from memory system.

B (m) or B (m, n): Number of requests that are serviced each module service time $T_s = T_c$, (m is the number of modules and n is number of requests each cycle.)

B (w) : Number of requests serviced per second.

$$B(w) = B(m) / T_s$$

Hellerman's Model

- One of the best known memory model.
- Assumes a single sequence of addresses.
- Bandwidth is determined by average length of conflict free sequence of addresses. (ie. No match in w low order bit positions where $w = \log_2 m$: m is no of modules.)
- Modeling assumption is that no address queue is present and no out of order requests are possible.

Hellerman's Model

- Under these conditions the maximum available bandwidth is found to be approximately.

$$B(m) = \sqrt{m}$$

$$\text{and } B(w) = \sqrt{m} / T_s$$

- The lack of queuing limits the applicability of this model to simple unbuffered processors with strict in order referencing to memory.

Strecker's Model

- Model Assumptions:
  - *n* simple processor requests made per memory cycle and there are *m* modules.
  - There is no bus contention.
  - Requests random and uniformly distributed across modules. Prob of any one request to a particular module is 1/m.
  - Any busy module serves 1 request
  - All unserviced requests are dropped each cycle
    - There are no queues

Strecker's Model

- Model Analysis:
  - *Bandwidth B(m,n) is average no of memory requests serviced per memory cycle.*
  - This equals average no of memory modules busy during each memory cycle.

  Prob that a module is not referenced by one processor = (1-1/m).

  Prob that a module is not referenced by any processor = $(1-1/m)^{n}$.

  Prob that module is busy = $1-(1-1/m)^{n}$.

  So B(m,n) = average no of busy modules

  = $\boldsymbol{m[1-(1-1/m)^{n}]}$

Strecker's Model

- Achieved memory bandwidth is less than the theoretical due to contention.
- Neglecting congestion carried over from previous cycles results in calculated bandwidth to be still higher.