3D-Stacked DRAM in Computer Memory Organization

Outline

• Introduction
• 3D-DRAM Architecture
• 3D-DRAM in Memory Systems
• 3D-DRAM as Last Level Cache
• 3D-DRAM as part of Main Memory
  • Chou et al. “CAMEO: A Two-Level Memory Organization with Capacity of Main Memory and Flexibility of Hardware-Managed Cache”. Micro 2015
• Conclusion
Introduction

What is 3D die stacking?

• 3D die stacking enables to stack multiple dies on top of one another in a 3D fashion
• Dies in different levels can be made of disparate Si technologies (e.g. CMOS and NMOS)
• Two different dies are connected through high bandwidth, low latency and low power consuming dense Through Silicon Vias (TSVs)
• Buses are replaced by TSVs

• Advantages:
  • Eliminates/reduce interconnection wire lengths within and across different dies
  • Reducing wire length results in reduced wire delay, reduced parasitic effect, lower area and power consumption

Introduction

What is 3D-DRAM?

• In 3D-stacked DRAM (3D-DRAM) multiple DRAM dies are stacked on top of bottom logic die
  ➢ The top-N layers store the actual data, i.e. DRAM bitcells
  ➢ The bottom logic layer stores the various control, access, and interface circuits
• Provides similar latency, 8x-16x bandwidth and ~70% dynamic energy/bit than conventional DDRx DRAM
• Two existing models
  ➢ Hybrid Memory Cube (HMC) (Intel’s Knights Landing)
  ➢ High Bandwidth Memory (HBM)
3D-DRAM Architecture: HMC

- Part of DRAM arrays from different layers form one vault
- Vault controller logic is implemented in the bottom logic base
- High-speed links are connected to the logic base

![HMC architecture figure]

3D-DRAM Integration

- In vertical stacking heat dissipation is challenging
- In interposer stacking the processor and 3D-DRAM are connected through high-speed links through silicon

![Possible integrations of 3D-DRAM figure]
3D-DRAM in Memory System

• Two possible placements have been investigated
  • 3D-DRAM as last level cache (LLC):
    • No change required at software level
    • Tag management of such large cache is challenging
      96 MB tag for 1 GB cache
  • 3D-DRAM as part of main memory (Flat-address-space system):
    • Provides more memory capacity
    • Placement of data in the faster memory is challenging

3D-DRAM as LLC

• 3D-DRAM as large Last Level Cache
  • Filter out accesses to off-chip main memory
  • Provides lower access latency and high bandwidth for the most commonly used data

• 2 types of die stacked DRAM cache
  • Block based, Page-Based
  • State-of-the-art approaches fail to address all the challenges
    • Scalability
    • Tag overhead
    • Hit ratio
    • Hit Latency
Research Overview

Block-Based caches
- 64 B blocks – tag overhead is high (1GB cache, 128MB tags)
- Store tags in DRAM instead SRAM
- Doesn’t exploit high spatial locality of server workloads – low hit ratio
- Miss predictor can add additional latency to hits

Page-Based caches
- 2 KB blocks (page) – maximized hit rates due to spatial locality
- Many blocks not accessed – wasted off-chip bandwidth
- Footprint prediction – fetch only those blocks which are going to be used
- Tags stored in SRAM – problem with scalability

Unison Cache

- **Unison Cache** - stacked DRAM cache design which leverages the best of block-based and page-based caches
  - Page-based
  - Low hit latency
  - High hit rates
  - Low tag overhead
  - Scalable
  - High effective capacity
  - Preserves off-chip bandwidth
Implementation/Methodology

- Unison Cache combines the best of Alloy Cache and Footprint Cache
  - **Alloy cache**
    - Block-based
    - Tags stored in DRAM, combined with data into Alloys for faster lookup
    - Low hit-rates – can’t exploit spatial locality
    - Direct mapped – low hit-rates
    - Miss prediction – can cause overfetch/underfetch
  - **Footprint cache**
    - Page-based
    - Tags stored in SRAM – scalability issue
    - Footprint prediction
      - (PC+offset) stored upon first access - trigger
      - Footprint vector updated on block access
      - Upon eviction store trigger and footprint vector in SRAM table

Unison Cache

- **Unison Cache**
  - Page-based
  - Tag in DRAM, tag and data are physically separated
    - Reading of the tag block and data block is overlapped, they work in unison
  - Footprint prediction
    - Footprint vectors are V/D bit vectors and stored next to the tags (metadata)
  - High hit-rates (90%) – no need for miss prediction
  - Set associative
    - Direct mapped increases misses because of higher conflict probability
    - All ways of the set are stored in the same DRAM row
    - Multiple sets can reside in a DRAM row
    - Way prediction to minimize lookup time and hit-latency
Implementation/Methodology

- Footprints are updated during block access by modifying V/D bits
  - Modified semantics of V/D bits to distinguish if a particular block has been accessed, so we can get the relevant blocks next time
  - **Overfetch** – block has been brought to cache, but not used
  - **Underfetch** – block has been requested but not found in the cache
  - Update the footprints accordingly and store them back in the predictor table upon eviction

- Singleton Prediction
  - Many footprints are only a single block – **singleton**
  - **Do not cache singletons**
  - If they are not in cache we can't update the footprint
  - Have a small singleton table to be able to update their footprint so we can cache them later
Implementation/Methodology

- Metadata causes alignment problems
  - Pages are now size of 960B instead of 1024B (1984B instead of 2048B)
  - Address mapping problems
  - Specialized logic for address manipulation
    - Authors estimate that it will take 2 cycles with several hundred gates (adders using residue arithmetic)

- Alternative cache designs
  - Block-based cache with footprint prediction
  - Page-based cache with tagged blocks

Results

Miss ratio comparison
- No need for more than 4 ways (figure not shown here)
- The small differences between FC and UC is due to different page sizes (2KB and 1KB respectively)
- AC has low hit rates due to low locality
- Large caches needed for large workloads such as TPC-H
- UC cache performs well
Results

<table>
<thead>
<tr>
<th>Predictor Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>• 1GB cache, 8GB for TPC-H</td>
</tr>
<tr>
<td>• For page-sized caches smaller pages are better in terms of overfetching</td>
</tr>
<tr>
<td>• Designs are bandwidth efficient – do not cause lot of overfetch</td>
</tr>
</tbody>
</table>

Performance

• As the cache size increases, pages stay longer in the cache exploiting locality
• With cache sizes, number of tags increase which harms FCs performance because of the SRAM lookup latency
• UC doesn’t have this problem because – tags in DRAM
Unison Cache Summary

• Unison Cache provides overall 14% improvement over AC design and 2% over FC
• Scalable, high-hit rates, low tag overhead
• Effective for large data centric workloads (such as TPC-H)

Limitation, Weaknesses, Scope of Improvement

• Indexing
  • Can we use the remaining 64 bytes per block for something else?
• Fragmentation
  • What is the fragmentation for page-based cache? A lot of blocks are not used, but pages need to be allocated.
• Larger Page Sizes
  • Are larger page sizes yield better performance for TPCH like applications?
• Alternative: Use 3D-DRAM as part of main memory
Flat-Address-Space System

- 3D-DRAM and DDRx DRAM together builds the main memory,
- **Objective:**
  - To place pages, that are going to be accessed in near future in 3D-DRAM to obtain better performance
- **Proposes:**
  - Epoch based schemes (epoch = 100 ms), where the OS transfers required pages among the memories so that the required pages are in 3D-DRAM

- **Hot-page policy**
  - Depending on a threshold number (e.g. 32) select which pages are hot
  - After each epoch the OS transfers the topmost hot pages to 3D-DRAM
- **First-touch policy**
  - At each epoch, whenever a page is first touched, transfer it to 3D-DRAM, if it is not already there

![Figure 1: Package-level organization of the target system consisting of a high-performance APU 2.5D-integrated with multiple DRAM stacks, along with conventional off-package DDR memory.](image)

Flat-address-space Management Technique: Hot-page policy

- **Hot-page policy**
  - Hot pages have accesses > threshold (32)
  - Cold pages have accesses < threshold (32)
  - Pages are transferred among HBM and PCM
  - Requires OS involvements

![Figure: Hot-page policy](image)
Flat-address-space Management Technique: Hot-page policy

- Results:
- Over a memory system composed of only DDR4 DRAM, FTHP (First-touch Hot-page) policy provides 15% speedup on average
- Consumes considerably less energy than other 3D-DRAM management techniques

![Graph: Speedup over DDR4 memory.](image)
![Graph: Total Memory Energy.](image)

Flat-Address-Space System
Chou et al. “CAMEO: A Two-Level Memory Organization with Capacity of Main Memory and Flexibility of Hardware-Managed Cache”. Micro 2045.

- 3D-DRAM and DDRx DRAM together builds a CCache-like MEmory Organization (CAMEO)
- **Objective:**
  - To place data in cache line granularity (64 bytes) in 3D-DRAM to obtain better performance
- **Proposes:**
  - Hardware based scheme that transfers data on-demand among the memories to get better performance while accessing the recently accessed data

![Diagram: CAMEO organization.](image)
Flat-address-space Management Technique:
CAMEO policy

- Four lines (each of 64-byte size) together form a “Congruence group”, one line is in 3D-DRAM and the rest in DDRx DRAM
- Only lines within the same Congruence group can swap places
- On a miss in 3D-DRAM required swapping takes place
- A Line Location Table (LLT) keeps track of updated locations of a line

Figure 4. Lines A, B, C and D form a Congruence Group. CAMEO performs swapping only within the Congruence Group.

Figure 5. Operation of the Line Location Table (LLT), which keeps location information for each Congruence Group. Lines A, B, C, and D form a Congruence group, and operation of LLT is shown after two memory requests are performed.

Flat-address-space Management Technique:
CAMEO policy

- Results:
  - CAMEO provides ~1.75x speedup over a system without any 3D-DRAM cache
  - CAMEO provides Energy Delay Product (EDP) of 0.5 when normalized to a system without any 3D-DRAM cache

Figure 13. Speedup with stacked memory. CAMEO outperforms both cache and Two Level Memory. CAMEO is close to an idealistic “DoubleUse” design that uses 4GB stacked memory as cache and also increases memory capacity by 4GB.

Figure 14. Comparison of Power and Energy-Delay Product. All the numbers are normalized to the baseline system.
Conclusion

• An active and promising research area:
  ➢ New generations of HMC and HBM are coming up
  ➢ Intel has already used HMC in their next generation Knights Landing processor
  ➢ Active research in cache and flat address space techniques

• Provides opportunity to place processing near to memory- Near Data Computing (NDC)

• 3D-DRAM is expected to serve a crucial role in satisfying the Department of Energy “Exascale System Challenge”