PENNSTATE

1855

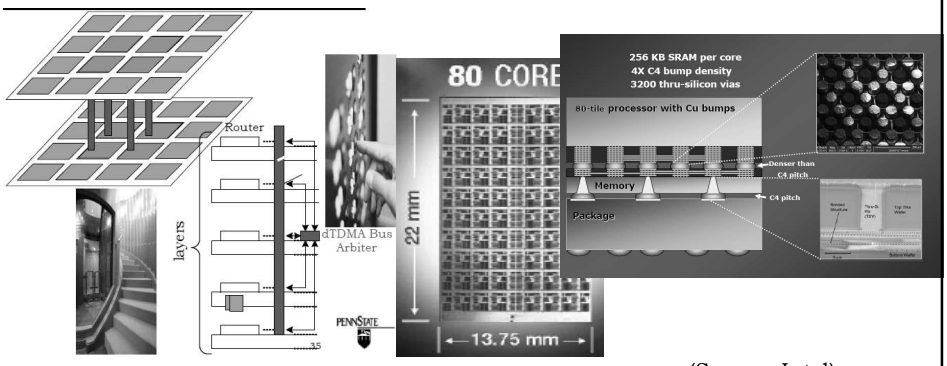
Memory-Stacking for Future High-Performance Microprocessor

Yuan Xie

Pennsylvania State University

3D Stacking Helps Future Many-Core Design

- **Provide fast access and high bandwidth**
  - see our paper [ISCA 2006]
  - Intel 80-core TeraFlop chip [ISSCC 2007]



(Source: Intel)

Business Model has Impacts on the Decision?

3D Stacked Microprocessor: Are We There Yet?

GABRIEL H. LOH
Georgia Institute of Technology
YUAN XIE
Pennsylvania State University

.....Three-dimensional integration has received considerable attention in the last several years from academic researchers and industry alike. This technology provides multiple layers of devices connected by a high-density, low-latency, layer-to-layer interface that can enable integrated circuits with more devices per unit area and allow the integration of different types of devices within the same 3D chip stack. Academic and technological leaders from a range of institutions, including major semiconductor companies, government agencies, and industry consortia. (Most respondents answered our questions on condition of anonymity, and some chose not to reply at all due to concerns over confidentiality and exposure of proprietary information.) Their responses provide a view of where 3D integration technology for microprocessors currently stands, Samsung, Tezzaron, and a few other companies have demonstrated, industry has reached the consensus that stacked memory will become mainstream. In this article, we focus on 3D stacking technology based on through-silicon-via (TSV) technology (see Figure 1b), which provides much faster and higher density inter-die connections than SiP or PoP.

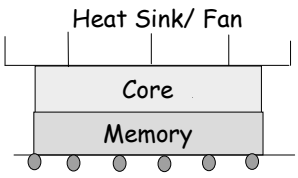
The first question that many people are interested in is simply when TSV-

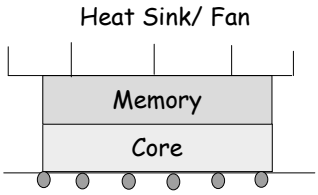
IEEE Micro 07/2010

Examples

(1) Intel's Tick-Tock Business Model

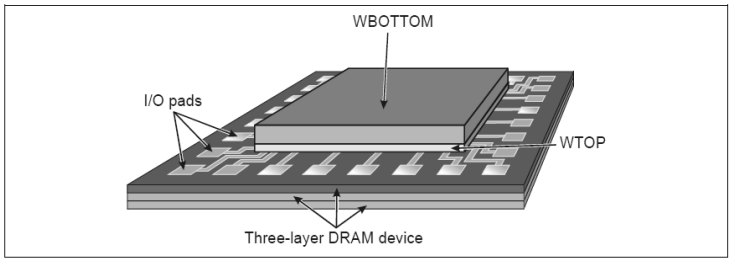
(2) Memory vs Logic



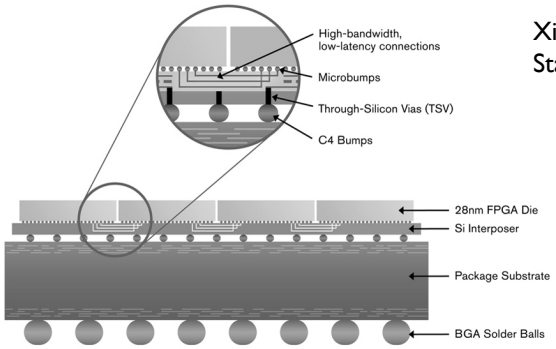


Wide-IO DRAM Stacking

- Demonstrate bandwidth benefits of 3D for future Quad High-Definition TV (HDTV) application --- first 3D IC prototyping of H.264 application
- Two logic layers (2.5x5mm^2)
  - WTOP & WBOTTOM
  - Micro-bump Connection
- Three DRAM layers (12.3x21.8mm^2) 256MB
- Chartered 130nm + Tezzeron TSV fabrication



A "More than Moore" Example



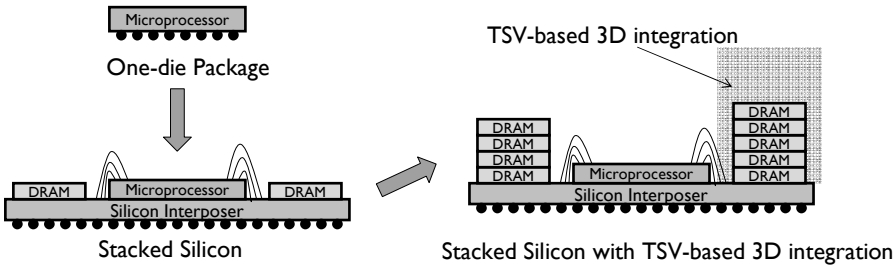
Xilinx®
Stacked Silicon Interconnect

"Enable 100x improvement in Die-to-Die Bandwidth Per watt"

"Enable 2-3x Capacity Advantage Over Monolithic Devices"

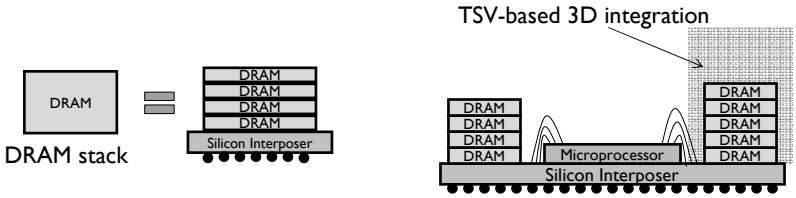
- Four FPGA dies inside one package
- Record 2 million logic cells
  - One logic cell = One 4-input LUT + One D-F/F (~500 transistors)
  - More than 1 billion transistors in a single package

Our Prospect: uP+Memory in Package



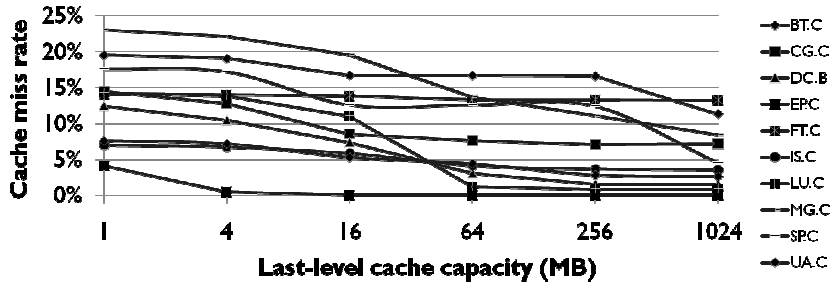
- More and more transistors can be integrated into a single package
- About 100MB-1GB on-package DRAM would be available
- How to use these transistors efficiently?
  - Multi-core, and many-core?
  - Larger cache size or deeper cache hierarchy?
  - On-package main memory?

Logic + Memory Integration



- Option 1: Use these DRAM stacks as caches
  - Processor chip cannot hold the tag array
    - Supposing 32 DRAM chips, the cache tags are equivalent to 2.1 DRAM
  - So, each DRAM chip holds both data and tag arrays
    - Supposing a 16-way associative cache, only 15-way is data, 1-way is tag
  - It is not energy-efficient to read out all of the15-way data
    - This means 2x cache access latency, one for tag, and one for data.

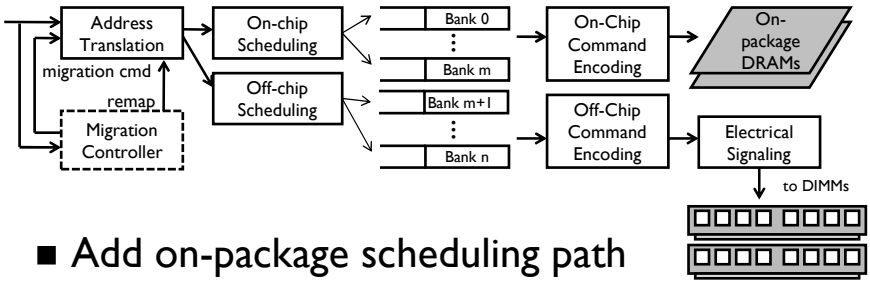
Use On-Package DRAM as Cache?



- Miss rate does not improve proportionally
- Hit latency and Miss penalty are close
  - Hit latency: On-package DRAM access (~70 cycles)
  - Miss penalty: Off-package DRAM access (~200 cycles)
- WARNING: Using on-package DRAM as cache might not be a good choice.

How to use on-package DRAM efficiently?

- Option 2: Use on-package DRAM as parts of the main memory



- Add on-package scheduling path
- Add migration controller to move data in and out from on-package DRAMs

Which One is Better? Preliminary Analysis

Last-Level Cache

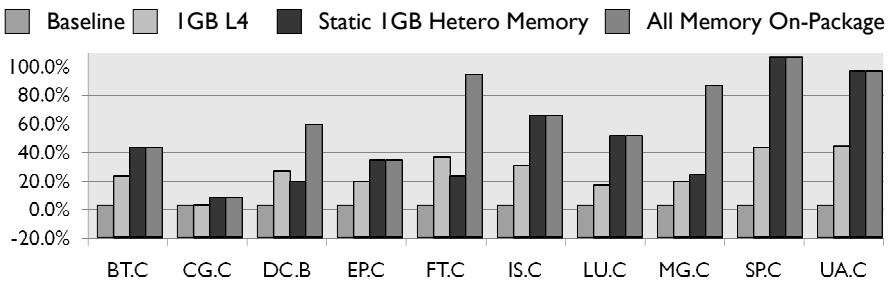
- Tag array overhead
- Cache access latency = 2x DRAM access latency
- Diminishing returns on miss rate
- Straightforward hardware control

Parts of Main Memory

- Need memory controller support
- Static mapping
  - Result in non-optimal data partitioning
- Dynamic mapping
  - Need data migration

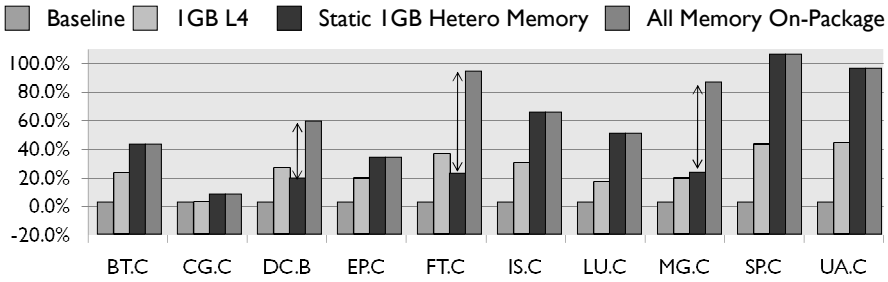
We first only consider the heterogeneous main memory with static mapping

LLC vs Static Heterogeneous Memory



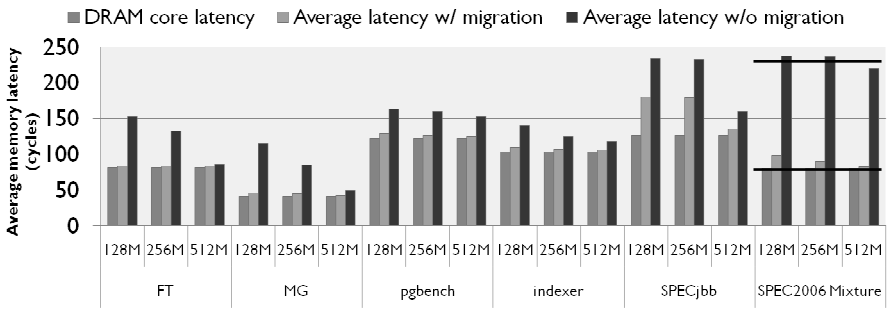
- Baseline:
  - 4-core 3.2Hz Nehalem-like processor
  - 32KB L1, 256KB L2, 8MB L3
- Add 1GB L4 cache:
  - Average performance improvement ~20%
- Ideal case (all memory on-package)
  - Average performance improvement ~60%
- Static mapping (1GB on-package memory)
  - When meeting the application footprint (<1GB) = ideal case
  - When not, it is still comparable to the case of 1GB L4 cache

Dynamic Hetero Main Memory



- Static hetero memory works good when application memory footprint is less than 1GB.
- If not, how to approach the performance of the ideal case?
- Solution: Dynamic data migration between on-package and off-package memory regions.

Data Migration Effectiveness



- After data migration, the average memory access latency is approaching the ideal case.

Conclusion

- Silicon Interposer provides a nice way to integrate 3D DRAM with processor.
- Using on-package DRAMs as last-level caches is not an efficient way in terms of performance
- Heterogeneous main memory (on-package DRAM and off-package DIMM) is promising
- Dynamic migration with on-chip memory controller support is important to the effectiveness of heterogeneous main memory

More details Please see our Supercomputing 2010 paper at http://www.cse.psu.edu/~yuanxie/3d.html