THERMAL-AWARE MEMORY MAPPING AND TASK ALLOCATION FOR 3D STACKED MEMORY AND PROCESSOR ARCHITECTURE

Wei-Kai Cheng
Dept. of ICE, Chung Yuan Christian University
OUTLINE

- Introduction
- Previous Work
- Observation
- Problem Formulation
- Algorithm and Program Flow
- Experiment
- Conclusion
INTRODUCTION

- 3D die stacking
  - Partition a system into multiple dies, then stacking these dies vertically to form a 3D IC
  - High density TSV (Through-Silicon Via) are inserted between stacking dies for the purpose of data communication channels

Courtesy: Borkar, Intel
INTRODUCTION

The advantages of 3D

- Reduction in interconnect wire length
- Improved memory bandwidth
- Realization of heterogeneous integration
- High packing density and small footprint
- Yield improvement for large die or new technology

S. Tiwari; “Potential, Characteristics, and Issues of 3D SOI; 3D SOI Opportunities”
Short Course, 2005 International SOI Conference
INTRODUCTION

- The challenges of 3D
  - Design tools and methodologies
  - Testing
  - Thermal management due to the increased power density
    - Cost for extra cooling system
    - Performance degradation
    - IR Drop
**Previous Work**


- Thermal Effect of 3D Memory Architecture

![Diagram of DRAM dies in a 3D package](image.png)

From 2D to 3D
PREVIOUS WORK

- Consider stacking effect and segment access frequency

```cpp
ExampleProgram()
{
    funcA(); // access segment A
    funcB(); // access segment B
    funcC(); // access segment C
    funcD(); // access segment D
}
```

<table>
<thead>
<tr>
<th>Segment</th>
<th>Access Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>funcA()</td>
<td>70%</td>
</tr>
<tr>
<td>funcB()</td>
<td>25%</td>
</tr>
<tr>
<td>funcC()</td>
<td>35%</td>
</tr>
<tr>
<td>funcD()</td>
<td>80%</td>
</tr>
</tbody>
</table>

![Diagram showing mapping and temperature vs execution stage]
PREVIOUS WORK

- Determination of candidate configurations
- ILP formulation for mapping optimization

Memory System

SIP Model

Floorplan of a DRAM chip

Memory Banks of Tiers

(a) : Configuration I
(b) : Configuration II
**PREVIOUS WORK**


- Sorting tasks according to power density
- Allocating super tasks to super cores

![Diagram of power density and super tasks](image-url)
**Previous Work**

- Algorithm for Homogeneous Floorplan

Sum up the temperature of super cores

Assign super tasks onto super cores
Previous Work

- Algorithm for heterogeneous floorplan

- Max_diff: \{P_1, P_2, \ldots, P_L\}, \{P_{L+1}, \ldots, P_{2L}\}, \ldots
- Mod_diff: \{P_1, P_{L+1}, P_{2L+1}, \ldots\}, \{P_2, P_{L+2}, \ldots\}, \ldots
- Min_diff: The principle is to balance the total powers of super tasks.

\[ \theta = \frac{\Delta T}{\Delta P} \]

Estimation eqn. for super tasks selection
OUTLINE

- Introduction
- Previous Work
- Observation
- Problem Formulation
- Algorithm and Program Flow
- Experiment
- Conclusion
**Observation**

- **Example: Segment Frequency of Task**
  - (Data Frequency + Instruction Frequency)

<table>
<thead>
<tr>
<th>Task</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>T4</th>
<th>T5</th>
<th>T6</th>
<th>T7</th>
<th>T8</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1:4</td>
<td>S1:6</td>
<td>S1:3</td>
<td>S1:4</td>
<td>S1:7</td>
<td>S1:2</td>
<td>S1:4</td>
<td>S1:1</td>
<td></td>
</tr>
<tr>
<td>S2:4</td>
<td>S2:6</td>
<td>S2:2</td>
<td>S2:3</td>
<td>S2:2</td>
<td>S2:2</td>
<td>S2:1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S3:3</td>
<td>S3:3</td>
<td>S3:1</td>
<td>S3:1</td>
<td>S3:2</td>
<td>S3:2</td>
<td>S3:1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S4:3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S4:1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S5:2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S5:1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S6:2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>S6:1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S7:1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S8:1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Frequency</td>
<td>20</td>
<td>15</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>
**Observation**

- **Observation 1: Task Allocation**
  - Balance the total frequency can balance the power density

<table>
<thead>
<tr>
<th>Core 1</th>
<th>Tasks</th>
<th>Total Segments</th>
<th>Total Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core 1</td>
<td>T1,T2</td>
<td>11</td>
<td>35</td>
</tr>
<tr>
<td>Core 2</td>
<td>T3,T4</td>
<td>9</td>
<td>17</td>
</tr>
<tr>
<td>Core 3</td>
<td>T5,T6</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>Core 4</td>
<td>T7,T8</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Core 1</th>
<th>Tasks</th>
<th>Total Segments</th>
<th>Total Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core 1</td>
<td>T1</td>
<td>8</td>
<td>20</td>
</tr>
<tr>
<td>Core 2</td>
<td>T3,T5</td>
<td>7</td>
<td>16</td>
</tr>
<tr>
<td>Core 3</td>
<td>T2,T8</td>
<td>6</td>
<td>18</td>
</tr>
<tr>
<td>Core 4</td>
<td>T4,T6,T7</td>
<td>7</td>
<td>18</td>
</tr>
</tbody>
</table>

- **Objective :** Balance the total frequency of CPU cores
- **Constraint :**
  Total segments for each core \( \leq \) Max segment number
Observations

- Observation 2: Task Scheduling
  - Avoid executing heavy tasks simultaneously

<table>
<thead>
<tr>
<th>Heavy</th>
<th>Light</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task1</td>
<td>Task5</td>
</tr>
<tr>
<td>Task2</td>
<td>Task6</td>
</tr>
<tr>
<td>Task3</td>
<td>Task7</td>
</tr>
<tr>
<td>Task4</td>
<td>Task8</td>
</tr>
</tbody>
</table>

![Diagram showing task scheduling with and without interlacing heavy and light tasks](image-url)
Observations

- Observation 3: Task Scheduling and Memory Mapping
  - Avoid all memory access from the same group

<table>
<thead>
<tr>
<th>Core 1</th>
<th>Task 1</th>
<th>Task 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core 2</td>
<td>Task 4</td>
<td>Task 8</td>
</tr>
<tr>
<td>Core 3</td>
<td>Task 2</td>
<td>Task 6</td>
</tr>
<tr>
<td>Core 4</td>
<td>Task 3</td>
<td>Task 7</td>
</tr>
</tbody>
</table>
OUTLINE

- Introduction
- Previous Work
- Observation
- Problem Formulation
- Algorithm and Program Flow
- Experiment
- Conclusion
**Problem Formulation**

- **Architecture**
  - 4 Cores in a layer
  - 4 Memory dies in a layer
  - 4 Banks in a memory die
  - M/N Groups (configurable)
PROBLEM FORMULATION

- **Input**
  1. Application Programs
  2. Reference Information

- **Output**
  1. Memory Segment Mapping
  2. Task Allocation
  3. Round Robin Task Scheduling

- **Objective**
  1. Minimal highest temperature after running all the applications
OUTLINE

- Introduction
- Previous Work
- Observation
- Problem Formulation
- Algorithm and Program Flow
- Experiment
- Conclusion
Applications analysis
- Behavior analysis
- Segment generation

Scheduling
- Multi-core tasks Scheduling

Memory block mapping
- Segment mapping
- Optimize mapping and scheduling

3D thermal behavior simulation
- HotSpot4.0
APPLICATIONS ANALYSIS

- Behavior analysis
  - Instruction count
  - Execution time
  - Memory address
  - Reference count

<table>
<thead>
<tr>
<th>sim: ** simulation statistics **</th>
</tr>
</thead>
<tbody>
<tr>
<td>sim_num_insn</td>
</tr>
<tr>
<td>sim_num_refs</td>
</tr>
<tr>
<td>sim_num_loads</td>
</tr>
<tr>
<td>sim_num_stores</td>
</tr>
<tr>
<td>Sim_cycle</td>
</tr>
</tbody>
</table>
**MULTI-CORE TASKS SCHEDULING**

- **Sorting**
  - Based on segment number & total frequency

- **Task Scheduling**
  - Assign each task to a core processor

---

**Flowchart: Multi-core Tasks Scheduling**

- **Memory reference information**
- **Segments generation for application 1**
  - .
  - .
- **Segments generation for application N**
- **Output error information**

1. Sort the data of all applications
2. Schedule application in a CPU core
3. Check if segments in application are less than or equal to remain segment space in core.
4. If yes, continue. If no, check if end of all application?
5. If yes, scheduling complete. If no, continue.

---

End of all application?
### Tasks Scheduling

#### Sorting
- Based on segment number firstly
- Based on total frequency secondly

#### Before Sorting

<table>
<thead>
<tr>
<th>Task</th>
<th>Task1</th>
<th>Task2</th>
<th>Task3</th>
<th>Task4</th>
<th>Task5</th>
<th>Task6</th>
<th>Task7</th>
<th>Task8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Seg Num</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Total Frequency</td>
<td>20</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>15</td>
<td>9</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

#### Sorting based on Segment Number

<table>
<thead>
<tr>
<th>Task</th>
<th>Task1</th>
<th>Task5</th>
<th>Task6</th>
<th>Task2</th>
<th>Task4</th>
<th>Task7</th>
<th>Task3</th>
<th>Task8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Seg Num</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total Frequency</td>
<td>20</td>
<td>9</td>
<td>15</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>4</td>
</tr>
</tbody>
</table>

#### Sorting based on Frequency Number

<table>
<thead>
<tr>
<th>Task</th>
<th>Task1</th>
<th>Task5</th>
<th>Task6</th>
<th>Task2</th>
<th>Task3</th>
<th>Task4</th>
<th>Task8</th>
<th>Task7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Seg Num</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Total Frequency</td>
<td>20</td>
<td>15</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>
**Tasks Scheduling**

- **Task Scheduling**
  - Balance execution time of all processor cores
  - Balance memory reference of all processor cores

- **Constraint:**
  - Total segments scheduled in each core \( \leq \) Max segment size for each core

- **Objective Function:**

\[
\begin{align*}
\min ( | & \text{Total} f_{\text{Core}_1} - \text{Total} f_{\text{Core}_2} | + \cdots ) \\
\min ( & \sum_{i=1}^{3} \sum_{j=1}^{4-i} | \text{Total} f_{\text{Core}_i} - \text{Total} f_{\text{Core}_{i+j}} | )
\end{align*}
\]
**Tasks Scheduling**

- **Task Scheduling**
  - **Assumption**: Max segment size for each core = 8

<table>
<thead>
<tr>
<th>Task</th>
<th>Task1</th>
<th>Task6</th>
<th>Task5</th>
<th>Task2</th>
<th>Task4</th>
<th>Task7</th>
<th>Task3</th>
<th>Task8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Seg Num</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total Frequency</td>
<td>20</td>
<td>9</td>
<td>15</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Core</th>
<th>Remain Segment Space</th>
<th>Total Frequency</th>
<th>Task Number</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core1</td>
<td>(8,0)</td>
<td>20</td>
<td>1</td>
<td>T1</td>
</tr>
<tr>
<td>Core2</td>
<td>(8,0)</td>
<td>16</td>
<td>2</td>
<td>T6,T3</td>
</tr>
<tr>
<td>Core3</td>
<td>(8,0)</td>
<td>18</td>
<td>2</td>
<td>T5,T7</td>
</tr>
<tr>
<td>Core4</td>
<td>(8,0)</td>
<td>18</td>
<td>3</td>
<td>T2,T4,T8</td>
</tr>
</tbody>
</table>

**Sorting based on segment number: 12**

<table>
<thead>
<tr>
<th>Core</th>
<th>Remain Segment Space</th>
<th>Total Frequency</th>
<th>Task Number</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core1</td>
<td>0</td>
<td>20</td>
<td>1</td>
<td>T1</td>
</tr>
<tr>
<td>Core2</td>
<td>1</td>
<td>16</td>
<td>2</td>
<td>T6,T3</td>
</tr>
<tr>
<td>Core3</td>
<td>2</td>
<td>18</td>
<td>2</td>
<td>T5,T7</td>
</tr>
<tr>
<td>Core4</td>
<td>1</td>
<td>18</td>
<td>3</td>
<td>T2,T4,T8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Core</th>
<th>Remain Segment Space</th>
<th>Total Frequency</th>
<th>Task Number</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core1</td>
<td>(8,0)</td>
<td>20</td>
<td>1</td>
<td>T1</td>
</tr>
<tr>
<td>Core2</td>
<td>(8,0)</td>
<td>16</td>
<td>2</td>
<td>T6,T3</td>
</tr>
<tr>
<td>Core3</td>
<td>(8,0)</td>
<td>18</td>
<td>2</td>
<td>T5,T7</td>
</tr>
<tr>
<td>Core4</td>
<td>(8,0)</td>
<td>18</td>
<td>3</td>
<td>T2,T4,T8</td>
</tr>
</tbody>
</table>

**Sorting based on frequency number: 26**

<table>
<thead>
<tr>
<th>Core</th>
<th>Remain Segment Space</th>
<th>Total Frequency</th>
<th>Task Number</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core1</td>
<td>0</td>
<td>20</td>
<td>1</td>
<td>T1</td>
</tr>
<tr>
<td>Core2</td>
<td>0</td>
<td>21</td>
<td>2</td>
<td>T5,T4</td>
</tr>
<tr>
<td>Core3</td>
<td>2</td>
<td>13</td>
<td>2</td>
<td>T6,T8</td>
</tr>
<tr>
<td>Core4</td>
<td>1</td>
<td>18</td>
<td>3</td>
<td>T2,T3,T7</td>
</tr>
</tbody>
</table>
**SEGMENT MAPPING**

- **Segment Separation**
- **Mapping Type Determination**
  - Round Robin Task Scheduling
- **Optimize Mapping and Scheduling**
  - Read Core record
  - Segment separate
  - Decide mapping type
  - End of all Cores
    - No
    - Yes
    - Optimize Mapping and Scheduling
**SEGMENT SEPARATE**

- **Segment separation**
  - Applied only if memory banks not fully utilized
  - High frequency segment is separated

- Read Core record
- Segment separate
- Decide mapping type
- End of all Cores
- Optimize Mapping and Scheduling

---

- Group 2
  - Core2
  - Memory banks not fully utilized

- Group 1
  - Core2
  - Memory banks fully utilized
SEGMENT MAPPING TYPE

- Mapping Type Determination
  - Two configuration types
  - Segments mapping to memory banks

- Read Core record
- Segment separate
- Decide mapping type
- End of all Cores
  - NO
  - Yes
    - Optimize Mapping and Scheduling
Consideration 1:
- Avoid direct Stacking in a segment
Consideration 2:
- Thermal dissipation of memory banks
  - Group that closer to heat sink has better thermal dissipation
  - Outer banks are better than inner banks for thermal dissipation
  - Thermal dissipation ability $1 > 2 > 3$
**Segment Mapping Type**

- **Segment mapping type 1:**
  - Applied when frequency difference between segments is large

![Diagram showing segment mapping type 1 with 4 dies in a tier.](image)
Segment Mapping Type

- Segment mapping type 2:
  - Applied when frequency difference between segments is small

```
Task  Seg1  Seg2  Seg3  Seg4
Group 1  1     2     3     4
        3     4     1     2
        4     2     1     3
        2     4     3     1
```

```text
Group 1

<table>
<thead>
<tr>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seg1</td>
</tr>
<tr>
<td>Seg2</td>
</tr>
<tr>
<td>Seg3</td>
</tr>
<tr>
<td>Seg4</td>
</tr>
</tbody>
</table>
```

heavy
light
Optimize Mapping and Scheduling

- Optimize Mapping and Scheduling
  - Optimize task execution order
  - Optimize task overlapping

1. Read Core record
2. Segment separate
3. Decide mapping type
4. End of all Cores
   - NO
   - Yes: Optimize Mapping and Scheduling
OPTIMIZE MAPPING AND SCHEDULING

- Optimize execution order
  - Round-robin task scheduling
  - Separate segments into sub-segments
  - Interlace sub-segments of different tasks
Objective: maximize total differential frequencies between successive sub-segments

ILP formulation for optimal execution order

\[
\text{Max}( \sum_{i=1}^{n-1} |f_i - f_{i+1}| )
\]
OPTIMIZE MAPPING AND SCHEDULING

- Optimize execution overlapping
  - Avoid overlapping of heavy sub-segments

- Insert idle slots between ASAP and ALAP time slots
  - Based on the longest execution time of all cpu cores
Evaluation:

- Analysis of Sub2-2
  - Weight of overlap is defined by distance of groups
  - Overlap with Sub1-2: 2 time unit
  - Overlap with Sub1-3: 2 time unit

- Evaluation: \( (4 - |1-1|) \times 2 + (4 - |3-1|) \times 2 = 12 \)

\[
\text{Min} \left[ \text{Group}_{\text{total}} - \left| \text{Group}_m^i - \text{Group}_{n}^{i+1} \right| \right] \times \text{Time}_{\text{execution}}
\]

- Total group
- Weight of overlap
- Overlap time
OPTIMIZE EXECUTION OVERLAPPING

- **ILP formulation for minimal overlapping**

```
\[
\begin{align*}
\text{ASAP} & : (4 - |2 - 2|)(3 - 0) + \\
& (4 - |1 - 2|)(8 - 3) + \\
& (4 - |1 - 1|)(10 - 8) + \\
& (4 - |3 - 1|)(12 - 10) + \\
& (4 - |3 - 3|)(14 - 12) + \\
& (4 - |4 - 3|)(22 - 14) = 71
\end{align*}
\]

\[
\begin{align*}
\text{ALAP} & : (4 - |2 - 2|)(3 - 0) + \\
& (4 - |1 - 2|)(8 - 3) + \\
& (4 - |1 - 1|)(10 - 9) + \\
& (4 - |3 - 1|)(13 - 10) + \\
& (4 - |4 - 3|)(24 - 14) = 67
\end{align*}
\]

\[
\begin{align*}
\text{ALAP} & : (4 - |2 - 2|)(3 - 2) + \\
& (4 - |1 - 2|)(10 - 3) + \\
& (4 - |3 - 1|)(14 - 10) + \\
& (4 - |4 - 3|)(24 - 14) = 63
\end{align*}
\]
```
OUTLINE

- Introduction
- Previous Work
- Observation
- Problem Formulation
- Algorithm and Program Flow
- Experiment
- Conclusion
EXPERIMENT

- Thermal simulation by HotSpot4.0
  - Power trace generation
  - Segment generation: average size vs. average frequency
  - Segment separation: separate vs. non separate
  - Task execution order: round-robin vs. no task swapping
  - Task overlapping: idle slot vs. no idle slot
OUTLINE

- Introduction
- Previous Work
- Observation
- Problem Formulation
- Algorithm and Program Flow
- Experiment
- Conclusion
CONCLUSION

- Integration of memory mapping and task scheduling for the 3D architecture
- Round-robin task scheduling for thermal optimization of CPU cores
- Idle slot insertion for thermal optimization of memory groups