Senior Thesis in Computer Engineering
University of Illinois at Urbana-Champaign
Advisor: Rakesh Kumar

Procurability-Aware Design Space Exploration

Lin Chuan Lee

Submitted in part fulfilment of the requirements for the degree of
Bachelor of Science in Computer Engineering of the
University of Illinois at Urbana-Champaign, May 2023

Abstract

Abstract goes here.

Contents

Chapter 1

Introduction

As the capability and reach of computing devices continue to rise, procurability becomes an important consideration. The semiconductor industry, with its unpredictable demand, complex supply chain, and high barrier of entry, is prone to shortages [1]. In addition to stockpiling inventory, observing the supply chain in advance, and other marketing strategies, companies using semiconductors can overcome supply chain disruptions by deploying engineering techniques to design products that rely less on difficult-to-procure components and use modular architectures that allow quick substitution of unavailable components [2]. For example, HBM memory offers higher bandwidth, lower power consumption, and a smaller form factor compared to DDR memory. However, HBM is less procurable because of a smaller number of suppliers and a higher integration cost due to the use of interposers. We performed a design space exploration to identify design points that can achieve acceptable performance and cost while replacing HBM with DDR memory by having different L3 capacities, number of memory channels, and memory frequency.

To perform the design space exploration, we constructed an analytical model of the system, including compute cores, cache, I/O, and memory controllers. The performance of each design point is derived from a roofline model that considers L3 hit rate, memory bandwidth, application profile, and compute core parameters. The cost of individual design points includes the cost of compute die, interposer, packaging, and memory. Using parameters from modern systems, we

observed regions where the system performance is bounded by L3 bandwidth, main memory bandwidth, and compute throughput. We identified design points where systems using DDR memory achieve the same performance as HBM memory at a lower cost by incorporating a larger L3 cache and a higher DDR frequency.

1.1 Section Title

Write here.

1.2 Thesis Outline

This thesis has been organized into six chapters. This section outlines the description of each chapter:

- In Chapter 2, we expose the fundamentals of ...
- In Chapter 3, we review in detail the state-of-the-art related to ...
- In Chapter 4, we present the methodology ...
- In Chapter 5, we describe ...
- In Chapter 6, we expose our final conclusions with future work possibilities.

Chapter 2

Background

Write here.

2.1 Section Title

Write here.

Chapter 3

Analytical Model

In this chapter, we describe the formulation of our analytical model to be used in design space exploration (DSE). The system, as modeled in the DSE, includes compute cores, shared L3 cache, memory controllers, and I/O (L1 and L2 caches are private to and part of each compute core). We use the term *memory subsystem* to refer to the memory hierarchy consisting of L3 and the main memory. The analytical model consists of three main parts. First, for each design point, the model estimates the performance via a roofline model that considers L3 cache hit rate, L3 bandwidth, main memory bandwidth, and compute throughput. Second, the model estimates the power and area of each design point to filter out unrealistic designs. Third, for each design point, the model estimates the cost of the compute die, memory, interposer (for HBM systems), and package.

Figures 3.1 and 3.2 show the cross-section of the physical organization of systems using DDR and HBM memory, respectively. HBM systems use a silicon interposer for 2.5D integration of the compute die and HBM stack. DRAM dies for DDR systems are located in off-chip DIMMs. Bump pitch and bump current rating values used in the model are labeled.

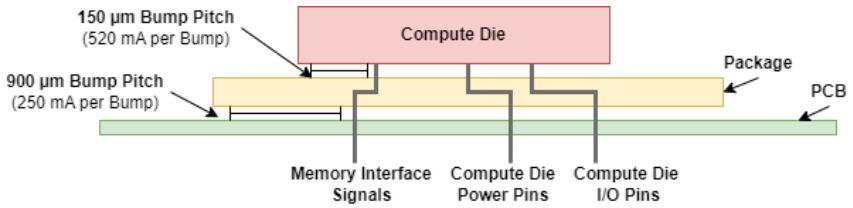

Figure 3.1: Cross-section of systems using DDR memory is shown, with bump pitch and bump current rating labeled.

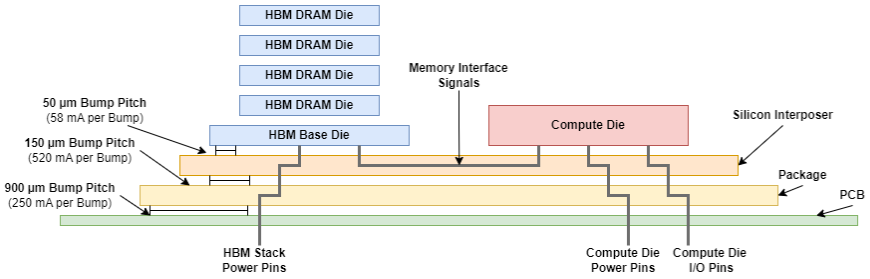

Figure 3.2: Cross-section of systems using HBM memory is shown, with bump pitch and bump current rating labeled.

3.1 Performance Model

Since the performance of the system is either compute-bound or memory-bound, we need to model both the compute throughput and the memory bandwidth of each design point. The compute throughput of the system is given by

$$\pi = CPI \cdot f_{\text{core}} \cdot N_{\text{core}} \tag{3.1}$$

where CPI is the cycles per instruction, f_{core} is the frequency of the compute cores, and N_{core} is the number of compute cores. We assume all compute cores operate at the same frequency.

The bandwidth of the memory subsystem is either bounded by the bandwidth between the compute cores and L3 or the bandwidth between L3 and the main memory. Hence, the memory subsystem bandwidth is given as

$$B_{\text{msys}} = \min\left(N_{\text{L3}} \cdot B_{\text{L3}}, \frac{N_{\text{MC}} \cdot B_{\text{MC}}}{\text{L3 Miss Rate}}\right) \tag{3.2}$$

where N_{L3} is the number of L3 slices, B_{L3} is the bandwidth per L3 slice, N_{MC} is the number of memory controllers, and B_{MC} is the bandwidth per memory controller. The first term is the bandwidth between compute cores and L3 while the second term is the bandwidth between L3 and main memory. The bandwidth between L3 and the main memory depends on the L3 miss rate because we assume that all memory accesses from the compute cores are serviced by L3, and only L3 misses are serviced by the main memory. A higher L3 hit rate implies a better utilization of the L3-main memory bandwidth. The L3 hit rate depends on the relative sizes of the application working set and L3. The hit rate scales linearly with the ratio of the L3 capacity to the working set size until the entire working set fits in L3, at which point we assume a nominal hit rate of 90%.

With the compute throughput and bandwidth of the memory subsystem given by Equations 3.1 and 3.2, respectively, the performance of the system is given as

$$Perf = \min\left(\pi,\ B_{\text{msys}} \cdot I_{\text{msys}}\right) \tag{3.3}$$

$$I_{\text{msys}} = \frac{Cap_{\text{workset}}}{Cap_{\text{workset}} - (Cap_{\text{L1}} + Cap_{\text{L2}})} \cdot I_{\text{app}} \tag{3.4}$$

where I_{msys} is the arithmetic intensity observed by the memory subsystem and I_{app} is the arithmetic intensity inherent to the applications. Since memory accesses are filtered by L1 and L2 caches before being serviced by the memory subsystem, the memory subsystem observes a higher arithmetic intensity than I_{app} for nonzero L1 and L2 capacities. We assume L1 and L2 cache to be exclusive of one another.

3.2 Enforcing Feasible Designs

To ensure all evaluated design points have reasonable power and area overhead, we developed models to estimate the power and area of each design point.

The thermal energy generated by the system can be dissipated either through the package

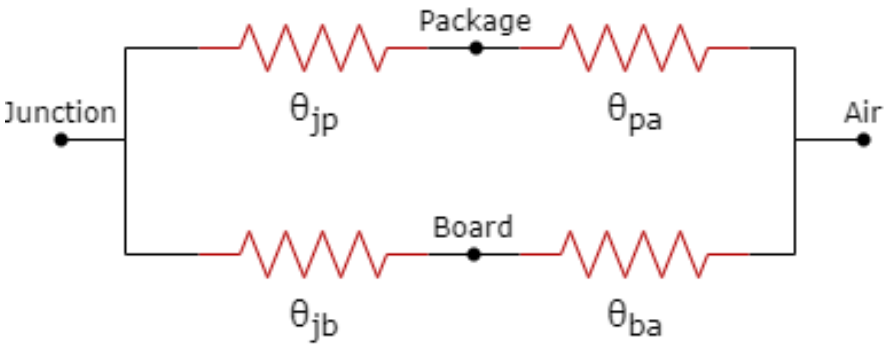

Figure 3.3: Thermal resistance network for the system.

casing or through the back of the PCB [3]. Based on the configuration of the thermal resistances shown in Figure 3.3, the system thermal resistance is given as

$$\theta_{\text{ja}} = \frac{(\theta_{\text{jp}} + \theta_{\text{pa}})(\theta_{\text{jb}} + \theta_{\text{ba}})}{\theta_{\text{jp}} + \theta_{\text{pa}} + \theta_{\text{jb}} + \theta_{\text{ba}}} \tag{3.5}$$

where θ_{jp}, θ_{pa}, θ_{jb}, and θ_{ba} denote the thermal resistance from junction to package, package to air, junction to PCB, and PCB to air, respectively. Assuming an ambient operating temperature of 25 $°C$ and a maximum junction temperature of 110 $°C$, the maximum power consumption of the system is given as

$$P_{\text{max}} = \frac{T_{\text{j, max}} - T_{\text{ambient}}}{\theta_{\text{ja}}} \tag{3.6}$$

Then, we compute the power consumption of the system to ensure it does not exceed P_{max}. First, the power of the compute cores is computed as $C \cdot V^2 \cdot f$ where C, V, and f are the capacitance, voltage, and frequency of the compute cores, respectively. The model also implements linear voltage-frequency scaling such that a reduction in frequency gives a cubic reduction in power. Second, the power of the memory controller is given as

$$P_{\text{MC}} = E_{\text{bit}} \cdot f_{\text{MC}} \cdot \#Wire_{\text{MC}} \cdot \left(\frac{V_{\text{MC}}}{V_{\text{MC, nominal}}}\right)^2 + \frac{f_{\text{MC}}}{f_{\text{MC, nominal}}} \cdot P_{\text{controller, nominal}} \tag{3.7}$$

where E_{bit} is the energy required to signal a bit from the compute die to the DRAM die,

f_{MC} is the frequency of the memory controller, $\#Wire_{\text{MC}}$ is the number of wires per memory controller, V_{MC} is the voltage of the memory controller. The first term is the physical power required to drive signals from the compute die to the DRAM die, and the second term is the power consumption of the memory controller logic. The memory controller operates at voltage and frequency values independent of the compute cores. The memory controller voltage also scales linearly with the memory controller frequency. Third, the power of the L3 cache is a linear function of L3 capacity. For systems with HBM memory, the power consumption of the DRAM stack also contributes to total system power consumption since the DRAM stack is integrated within the package.

The area of the system is computed as follows. We scale the area of the computing cores linearly with core frequency when the frequency exceeds some baseline which is a model parameter. This is important to account for the microarchitecture changes required to enable higher operating frequency. A 5% increase in core frequency results in a 10% increase in compute core area, with the exception that L1 and L2 area is only scaled by 2%. The final die area is the dot product between the vector of component counts and the vector of component areas.

Since bumps for the memory interface, I/O (PCIe lanes), and power delivery need to fit on the bottom of the die, the bumps enforce a lower bound on the die area. The lower bound is given as

$$A_{\text{die}} \geq (\text{Bump Pitch})^2 \cdot \left(\#Bump_{\text{power}} + \#Bump_{\text{MC}} \cdot N_{\text{MC}} + \#Bump_{\text{I/O}}\right) \tag{3.8}$$

$$\#Bump_{\text{power}} = \frac{P_{\text{die}}}{V_{\text{die}} \cdot I_{\text{bump}}} \cdot 2 \tag{3.9}$$

where $\#Bump_{\text{power}}$ is the number of bumps for power delivery to the die, $\#Bump_{\text{MC}}$ is the number of bumps for each memory controller, $\#Bump_{\text{I/O}}$ is the number of bumps for I/O, P_{die} is the power of the die, V_{die} is the voltage of the die, and I_{bump} is the current rating of each bump. If the die area is too small to fit all the bumps, we allocate as much dead space on the die as required. For systems using DDR memory, we assume a compute die to package bump

pitch of 150 μm with a current rating of 520 mA (see Figure 3.1). For systems using HBM memory, we assume a compute die to interposer bump pitch of 50 μm with a current rating of 58 mA (see Figure 3.2). Current rating per bump scales with bump pitch squared. For the upper bound on die area, we only consider systems with die areas less than 1000 mm^2 in our DSE.

Assuming the die has a 3:2 aspect ratio, the die area must be sufficiently large such that there is enough length of the perimeter of the die for wires. The model assumes wires are only connected to the two longer sides of the die. The maximum number of wires that can be connected to the die perimeter is given by

$$\#Wire_{\text{max}} = \left(6 \cdot \sqrt{\frac{A_{\text{die}}}{6}}\right) \cdot \text{PCB Layer} \cdot \text{Link Pitch} \tag{3.10}$$

where the expression in the parenthesis is the combined length of the two longer sides of the die. Since the memory interface accounts for most of the wires that connect to the die perimeter, we require $\#Wire_{\text{max}} \geq \#Wire_{\text{MC}} \cdot N_{\text{MC}}$ where $\#Wire_{\text{MC}}$ is the number of wires per memory controller and N_{MC} is the number of memory controllers.

3.3 Cost Model

To meaningfully perform DSE, we must model the cost associated with each design point. The total cost is the sum of the cost of compute die, memory, interposer (for HBM systems), and package.

To estimate the cost of the compute die that includes compute cores, L3, memory controllers, and I/O, we must model the yield. Since the impact on yield for logic versus memory is different, we use CACTI [4] to estimate the percentage of L1, L2, and L3 area that is dedicated to SRAM cell versus peripheral circuitry, which enables us to accurately define the portion of die area that will significantly impact yield. Then, the yield is given by

$$Y_{\text{die}} = \left(1 + \frac{A_{\text{die, yield}} \cdot D_0}{\alpha}\right)^{-\alpha} \tag{3.11}$$

where $A_{\text{die, yield}}$ is the die area for yield calculation purposes, D_0 is the defect density, and α is the clustering factor [5]. Then, we estimate the number of dies that can be cut from a wafer as

$$\#Die_{\text{wafer}} = d \cdot \pi \left(\frac{d}{4 \cdot A_{\text{die}}} - \frac{1}{\sqrt{2 \cdot A_{\text{die}}}}\right) \tag{3.12}$$

where d is the wafer diameter (300 mm) and A_{die} is the die area [6]. Finally, the cost per die is given as $\frac{Cost_{\text{wafer}}}{\#Die_{\text{wafer}} \cdot Y_{\text{die}}}$ where $Cost_{\text{wafer}}$ is the wafer cost.

The cost of interposers, for systems using HBM memory, is similarly computed using Equations 3.11 and 3.12 with die area replaced with interposer area. The interposer area is the sum of the die area and the HBM DRAM stack area. We also include the cost of assembling the die and the interposer.

For the cost of the package, we first estimate the area of the package. The package area is similarly computed using Equations 3.8 and 3.9. As shown in Figures 3.1 and 3.2, for systems using either DDR or HBM memory, we assume a package to PCB bump pitch of 900 μm with a current rating of 250 mA [3]. For HBM systems, the $(\#Bump_{\text{MC}} \cdot N_{\text{MC}})$ term is excluded from Equation 3.8 since the memory interface wires connecting the HBM memory controller on the compute die with the HBM DRAM stack lie within the interposer without exiting the package. For HBM systems, the P_{die} in Equation 3.9 also includes the power consumption of the DRAM stack since the DRAM stack is on-chip. With the package area computed, the package cost is estimated by assuming the cost-to-area ratio of \$0.02 per mm^2.

With memory prices from commercially available products [7, 8, 9, 10], we compute the total cost of the system as the sum of compute die, memory, interposer (for HBM systems), and package costs.

Chapter 4

Experimental Setup

With the analytical model described in Chapter 3, the next step is to establish the set of design points of interest to explore in the DSE.

The compute core parameters are based off of the Intel Xeon Platinum 8380 CPU. Since we are interested in the performance and cost tradeoff between HBM and DDR memory, a high throughput CPU will ensure most design points will not be compute-bound. For the area of L1, L2, and L3 cache, we assume an L3 cache density of 0.5 MB/mm^2 with L2 and L1 cache being 1.5x and 4x less dense to account for the larger peripheral circuitry. We verified the area of the L1, L2, and L3 cache using CACTI [4] with cache organization parameters based on an Intel Skylake processor [11]. Since the Intel Xeon Platinum 8380 CPU uses 10 nm process, we estimate the wafer cost based on TSMC's pricing for 300 mm wafers [12].

For L3 cache, we consider L3 capacity from 2 MB to 200 MB with a step size of 2 MB (each L3 slice is 2 MB). For memory, we consider a number of DDR and HBM memories with different frequencies and channel counts (see Table 4.1).

Then, the set of design points is the full factorial combination of different L3 capacities, memory types, memory frequencies, and memory channel counts. We evaluate each of these design points under different combinations of arithmetic intensities and working set sizes (see Table 4.2). Arithmetic intensity determines the regions in which the system is compute bound

Table 4.1: Memory Types Considered for DSE

Memory Type	**Frequency (MHz)**	**Channel Counts**
DDR4	2400, 3200	4, 6
DDR5	4800, 5600	4, 6
HBM2	1000	4

Table 4.2: Application Profiles Considered for DSE

Arithmetic Intensity (FLOPs/byte)	0.125, 0.25, 0.5, 1
Working Set Size (MB)	25, 50, 100, 150

versus bandwidth bound. Working set size determines (i) the degree to which memory accesses from the core are filtered by L1 and L2 cache before being serviced by the memory subsystem and (ii) the hit rate of the L3 cache in the memory subsystem.

Chapter 5

Results

In this chapter, we describe the results of applying our analytical model to the set of design points. We start by examining the attainable performance of design points. Then, we incorporate the cost model to identify design points with reasonable performance and cost.

5.1 Compute Bound versus Bandwidth Bound

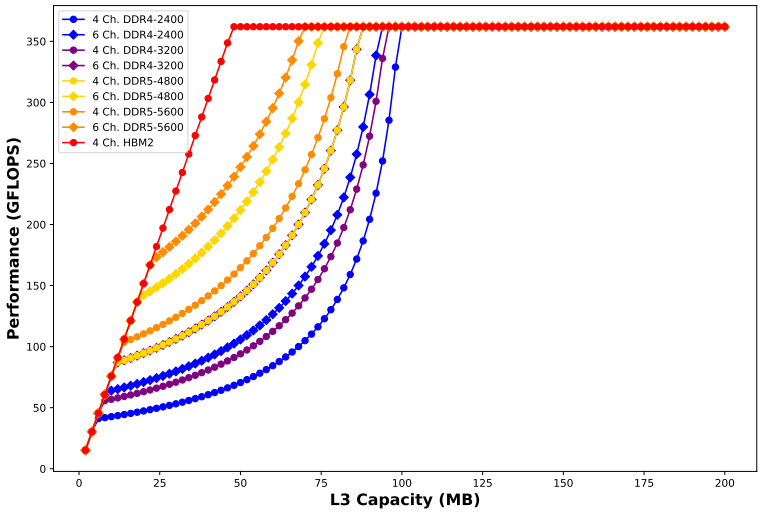

Figure 5.1: Line curve showing overall system performance versus L3 capacity. We show the curves for different memory types, frequencies, and channel counts. The plot is drawn for application arithmetic intensity of 0.5 FLOPs/byte and working set size of 100 MB.

Figure 5.1 shows the system performance of design points as a function of L3 capacity. A curve is drawn for a different combination of memory frequency and channel count. For the same L3 capacity, systems with higher memory frequency or more memory channels have better performance, as expected. Systems with HBM2 memory outperform, or have identical performance, compared to any DDR systems for any L3 capacity. To identify regions where the system performance is bounded by compute throughput, compute core-L3 bandwidth, or effective L3-main memory bandwidth, we have to examine the relationship between each of these quantities.

Figure 5.2 shows, as a function of L3 capacity, the bandwidth between compute core and L3 and the effective bandwidth between L3 and main memory for 4 channels of DDR5-4800 memory. The compute core-L3 bandwidth increases linearly as L3 capacity increases since adding additional slices of L3 cache increases the number of R/W ports. The effective L3-main memory bandwidth increases super-linearly because as L3 capacity increases for fixed working set size, the L3 hit rate increases, and the effective L3-main memory bandwidth has a $\frac{1}{1-x}$ dependence on the L3 hit rate (see Equation 3.2). However, the effective L3-main memory bandwidth stops increasing for L3 capacity larger than 100 MB because at that point the entire working set fits into the L3 cache.

Figure 5.3 shows, as a function of L3 capacity, the effective memory subsystem bandwidth and compute throughput also for 4 channels of DDR5-4800 memory (again, *memory subsystem* refer to the memory hierarchy consisting of L3 and the main memory). As shown in Equations 3.2 and 3.3, the effective memory subsystem bandwidth curve is the minimum of the two curves in Figure 5.2 multiplied by the arithmetic intensity. The compute throughput curve is constant since compute core parameters are not varied during the DSE. The minimum of the effective memory subsystem bandwidth and compute throughput curves gives the system performance curve for 4 channels of DDR5-4800 memory that was shown in Figure 5.1.

Based on these observations, we see that the system with 4 channels of DDR5-4800 memory is bounded by compute core-L3 bandwidth for L3 capacities from 2 MB to 12 MB. As we continue to add more slices of L3, the system performance becomes bounded by the effective

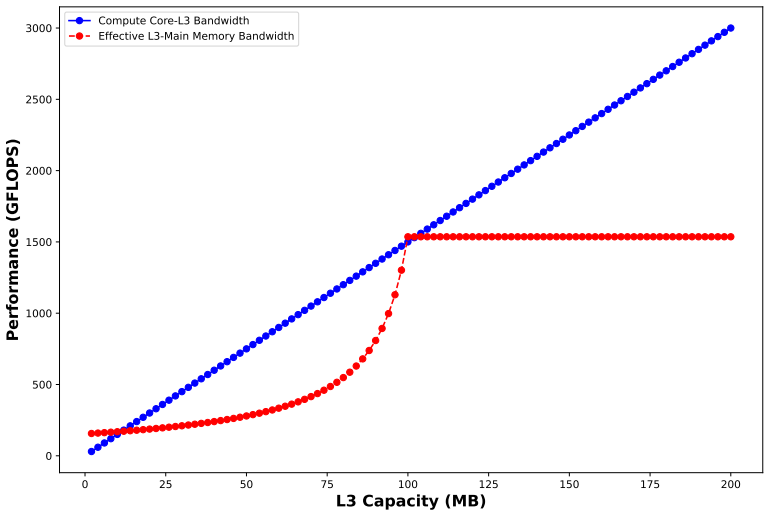

Figure 5.2: Line curve showing compute core-L3 bandwidth and effective L3-main memory bandwidth versus L3 capacity for 4 channels of DDR5-4800 memory. The plot is drawn for application arithmetic intensity of 0.5 FLOPs/byte and working set size of 100 MB.

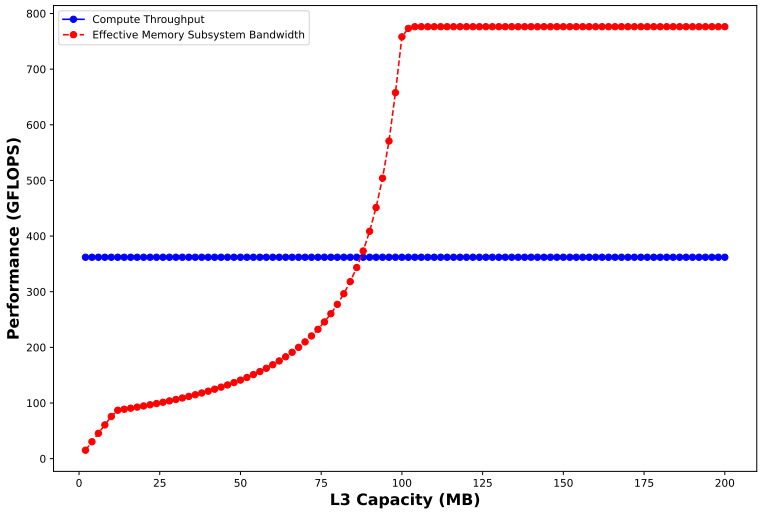

Figure 5.3: Line curve showing compute throughput and effective memory subsystem bandwidth versus L3 capacity for 4 channels of DDR5-4800 memory. The plot is drawn for application arithmetic intensity of 0.5 FLOPs/byte and working set size of 100 MB.

L3-main memory bandwidth. This is the case for L3 capacities from 12 MB to 88 MB. Further adding L3 slices increases L3 hit rate and effective L3-main memory bandwidth until the system performance becomes bounded by the compute throughput of the compute cores. This is true for L3 cache capacity greater than 88 MB. The relative positioning of the compute core-L3

bandwidth, effective L3-main memory bandwidth, effective memory subsystem bandwidth, and compute throughput curves change for different memory types, frequencies, and channel counts. In other words, for each memory configuration, the regions in which system performance is bounded by compute core-L3 bandwidth, effective L3-main memory bandwidth, and compute throughput differ. For instance, as shown in Figure 5.1, the system using 6 channels of DDR5-5600 memory has a smaller interval of L3 capacity during which the system performance is bounded by effective L3-main memory bandwidth since it has higher memory channel count and frequency.

5.2 Cost Composition

Next, we can understand the cost composition of each design point by examining the compute die, memory, interposer (for HBM systems), and package cost separately. We will compare the cost for systems with different memory configurations at iso-L3 capacity since the effect of adding an additional slice of L3 is the same across memory configurations. The effect of adding one additional slice of L3 is as follows. Compute die and interposer area (for HBM systems) increase marginally which lowers yield for compute die and interposer. It also decreases the number of compute dies and interposers that can be cut from a wafer. Consequently, compute die and interposer costs increase. Since a slightly larger L3 cache consumes more power, the area and cost of the package also increase marginally.

Figure 5.4 shows the compute die, memory, interposer, package, and total cost of systems with different memory configurations at iso-L3 capacity of 60 MB. We observe that the cost of compute die is greater for systems with six memory channels compared to four since the additional channels require additional memory controllers on the compute die, which takes up additional area on the wafer, reduces yield, and lowers the number of dies that can be harvested from a wafer. We see that memory cost also increases with the number of channels and frequency with HBM memory costing the most. Next, we consider the package cost. The package cost increases with the channel count since additional memory controllers require additional power

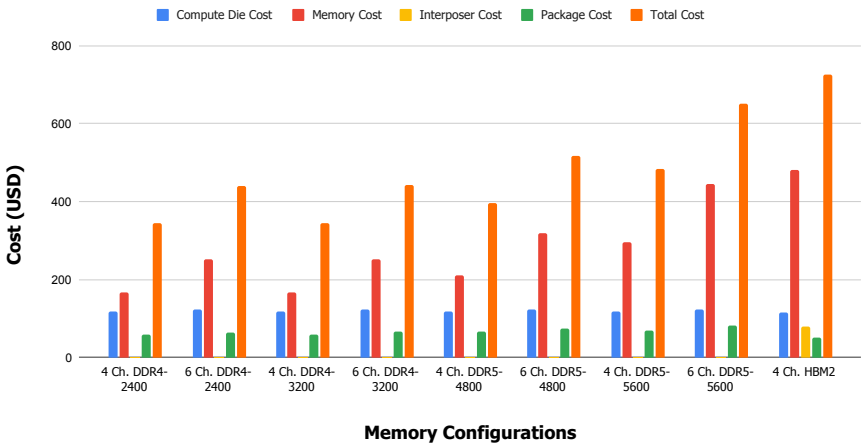

Figure 5.4: Compute die, memory, interposer, package, and total cost of systems with different memory configurations. All systems plotted have L3 capacity of 60 MB.

and additional bumps on the package to transmit data to off-chip DRAM dies. The package cost also increases with frequency since the memory controller's operating voltage scales linearly with frequency. We observe that the system with HBM2 memory has the cheapest package. This is because HBM's wide interface enables it to operate at a frequency lower than any DDR memories we considered while still achieving the highest bandwidth (we assume an operating frequency of 1000 MHz for HBM2). Since the HBM stack is on-chip compared to the off-chip DDR DIMM, the amount of energy required for the memory controller to signal a bit of data is approximately 4.29x @@(15pJ/b divide 3.5pJ/b) less. Finally, the integration of the DRAM stack within the package means memory interface wires connecting the HBM memory controller with the DRAM stack do not exit the package, which means fewer package-to-PCB bumps for signaling.

Overall, the system using HBM2 memory has the highest total cost with compute die, memory, interposer, and package accounting for 15.77%, 66.14%, 10.96%, and 7.14% of the total cost, respectively @@. The high cost of the HBM system relative to DDR systems is mostly attributed to (i) the inherently high cost of the HBM DRAM stacks which require the use of through-silicon vias and micro bumps compared to the widely available DRAM DIMMs and (ii) the high cost of interposer fabrication and assembly. In fact, in our HBM2 system with 60 MB of L3 cache, the interposer cost, including assembly, is approximately 69.49% of the compute die cost @@.

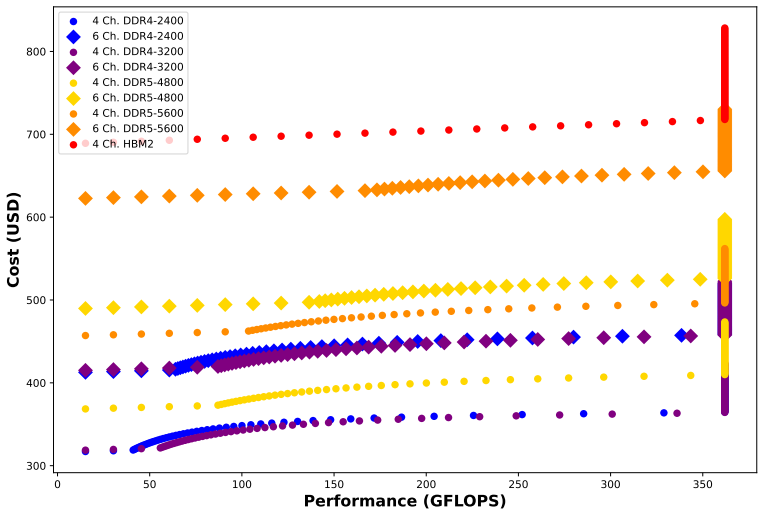

Figure 5.5: Scatter plot showing overall system cost versus performance. We show the data points for different memory types, frequencies, and channel counts. The plot is drawn for application arithmetic intensity of 0.5 FLOPs/byte and working set size of 100 MB.

5.3 Cost-Performance Trade-off

Now that we have examined the performance behavior and cost composition of the design points, we can evaluate the cost and performance trade-off. Figure 5.5 shows overall system cost versus performance where each design point is represented by a marker on the plot. For a given memory configuration (represented as a unique combination of marker color and shape), the leftmost marker is the design point with 2 MB of L3 cache while the rightmost marker is the design point with 120 MB of L3 cache. The vertically stacked markers on the far right represent systems that are compute-bound, at which point adding additional L3 slices only increases cost without improving performance.

From Figure 5.5, we can see that the same performance can be achieved using different combinations of L3 capacity, memory type, memory channel count, and memory frequency with very different total. For instance, Table 5.1 shows, for each memory configuration, the L3 capacity required to achieve 200 GFLOPS of performance (under this specific application profile of 0.5 FLOPs/byte arithmetic intensity and 100 MB working set) and the associated total cost. Therefore, the optimal cost configuration is to use 4 channels of DDR4-3200 memory

Table 5.1: L3 Capacity Required for 200 GFLOPS Performance for All Memory Configurations (Sorted by Total Cost)

Memory Configuration	L3 Capacity (MB)	Total Cost (USD)
4 Ch. DDR4-3200	82	357.08
4 Ch. DDR4-2400	90	359.58
4 Ch. DDR5-4800	68	399.78
6 Ch. DDR4-3200	68	447.18
6 Ch. DDR4-2400	78	449.93
4 Ch. DDR5-5600	60	484.37
6 Ch. DDR5-4800	46	510.78
6 Ch. DDR5-5600	36	638.72
4 Ch. HBM2	26	703.90

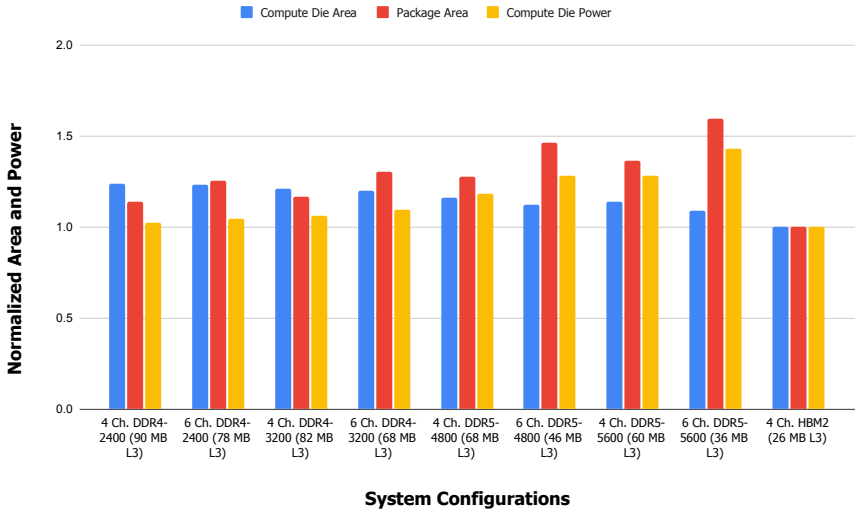

Figure 5.6: Compute die area, package area, and compute die power consumption (excluding power of HBM DRAM stack) for each design point shown in Table 5.1 normalized against the system with 4 channels of HBM2 memory and 26 MB of L3 cache.

with 82 MB of L3 cache. Other system configurations may not have the best performance and cost trade-off but may be superior in other metrics such as area and power consumption. For each system configuration in Table 5.1, Figure 5.6 compares the compute die area, package area, and compute die power consumption normalized against the system with 4 channels of HBM2 memory and 26 MB of L3 cache. This HBM system, despite having the highest total cost, outperforms all other DDR systems in Table 5.1 in compute die area, package area, and compute die power. This makes HBM a high-cost, low-procurability solution for systems with strict power and area constraints. Although Table 5.1 and Figure 5.6 are based on a target performance of 200 GFLOPS, the same conclusion, in fact, generalizes to all performance

thresholds shown in Figure 5.5.

Since the attainable performance of a given system configuration depends significantly on the application profile, namely the arithmetic intensity and working set size, we next see if the same conclusion generalizes to different application profiles. We reduce arithmetic intensity from 0.5 to 0.125 FLOPs/byte and increase the working set size from 100 MB to 150 MB to consider a more memory-intensive application profile. A larger working set implies a lower L3 hit rate and less filtering from L1 and L2 cache. Figure 5.7, similar to Figure 5.1, shows the system performance of design points versus L3 capacity. A line is drawn for a different combination of memory frequency and channel count. With a more memory-intensive application, only systems with 4 channels of HBM2 memory and more than 192 MB of L3 cache are able to fully utilize the compute throughput of the compute cores. For systems with any other memory configurations, the system performance eventually becomes bounded by the effective L3-main memory bandwidth. We also see alternating intervals of L3 capacity where the system performance is bounded by compute core-L3 bandwidth and effective L3-main memory bandwidth. For instance, the system with 6 channels DDR5-5600 memory is bounded by compute core-L3 bandwidth from 2 MB to 20 MB, bounded by effective L3-main memory bandwidth from 20 MB to 146 MB, bounded by compute core-L3 bandwidth from 146 MB to 178 MB, and finally bounded by effective L3-main memory bandwidth for L3 cache larger than 178 MB.

Figure 5.8, similar to Figure 5.5, shows overall system cost versus performance where each design point is represented by a marker on the plot. Similar to the case with 0.5 FLOPs/byte arithmetic intensity and 100 MB working set size, we observe that HBM systems have the highest total cost for all performance thresholds. Our model also shows that HBM systems have the lowest compute die area, package area, and compute die power for all performance thresholds compared to any systems using DDR memory. However, for the more memory-intensive application profile, there are performance levels not achievable using DDR memories regardless of channel count and frequency, namely performance greater than 338 GFLOPS.

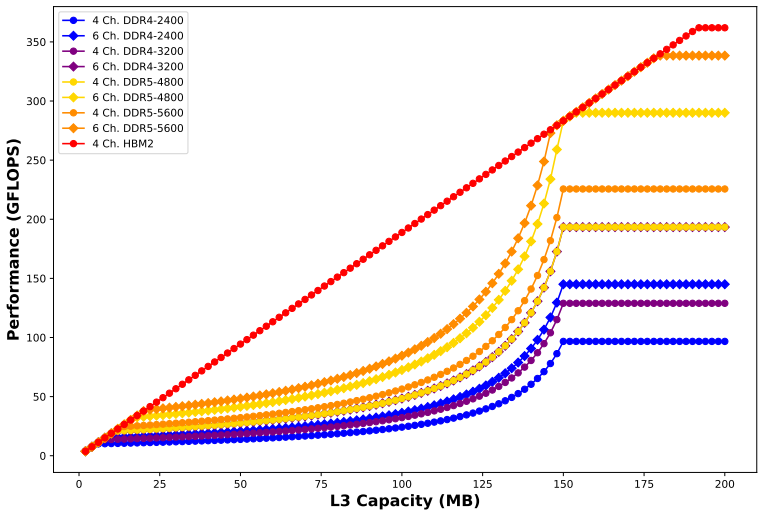

Figure 5.7: Line curve showing overall system performance versus L3 capacity. We show the curves for different memory types, frequencies, and channel counts. The plot is drawn for application arithmetic intensity of 0.125 FLOPs/byte and working set size of 150 MB.

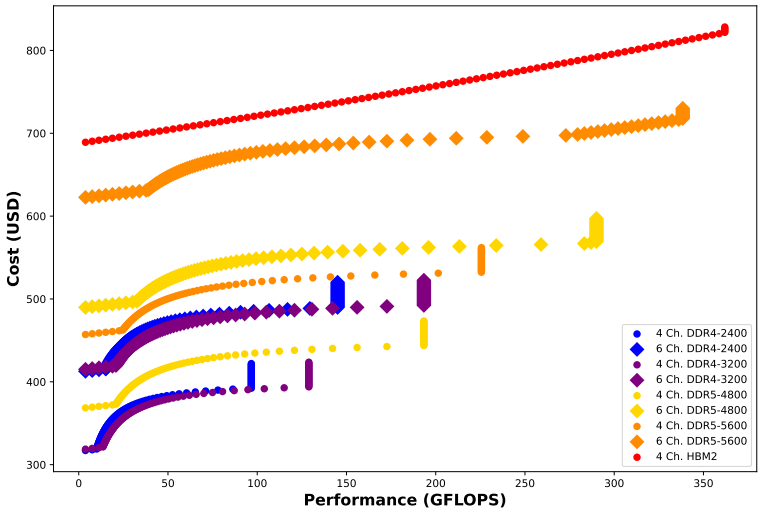

Figure 5.8: Scatter plot showing overall system cost versus performance. We show the data points for different memory types, frequencies, and channel counts. The plot is drawn for application arithmetic intensity of 0.125 FLOPs/byte and working set size of 150 MB.

Chapter 6

Conclusion

This chapter presents the conclusions of this thesis along with the advantages, disadvantages, and limitations of the methods. Future extensions and research are also included.

6.1 Summary of Thesis Achievements

Write here.

6.2 Future Work

Write here.

Bibliography

[1] R. Kumar, "We're going to see another chip shortage— despite the chips and science act," March 2023. [Online]. Available: fortune.com/2023/03/11/chips-and-science-act-semiconductor-shortage-rakesh-kumar/

[2] P. Hanbury, B. Radzevych, and B. Grant, "Engineering your way out of the global chip shortage," December 2021. [Online]. Available: hbr.org/2021/12/engineering-your-way-out-of-the-global-chip-shortage

[3] S. Pal, D. Petrisko, A. A. Bajwa, P. Gupta, S. S. Iyer, and R. Kumar, "A case for packageless processors," in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2018, pp. 466–479.

[4] S. Wilton and N. Jouppi, "Cacti: an enhanced cache access and cycle time model," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 5, pp. 677–688, 1996.

[5] S. Pal, D. Petrisko, R. Kumar, and P. Gupta, "Design space exploration for chiplet-assembly-based processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 4, pp. 1062–1073, 2020.

[6] AnySilicon, "Die per wafer (free) calculator." [Online]. Available: anysilicon.com/die-per-wafer-formula-free-calculators/

[7] Crucial, "Crucial 16gb ddr4-2400 udimm." [Online]. Available: crucial.com/memory/ddr4/ct16g4dfd824a

[8] ——, "Crucial 16gb ddr4-3200 udimm." [Online]. Available: crucial.com/memory/ddr4/ct16g4dfra32a

[9] ——, "Crucial 16gb ddr5-4800 udimm." [Online]. Available: crucial.com/memory/ddr5/ct16g48c40u5

[10] ——, "Crucial 16gb ddr5-5600 udimm." [Online]. Available: crucial.com/memory/ddr5/ct16g56c46u5

[11] WikiChip, "Skylake (server) - microarchitectures - intel," May 2017. [Online]. Available: en.wikichip.org/wiki/intel/microarchitectures/skylake_(server)

[12] S. M. Khan and A. Mann, "Ai chips: What they are and why they matter," *Center for Security and Emerging Technology*, pp. 44–45, April 2020.