EquiNox: Equivalent NoC Injection Routers for Silicon Interposer-based Throughput Processors

Yunfan Li, Lizhong Chen
School of Electrical Engineering and Computer Science
Oregon State University, Corvallis, USA
{liyunf, chenliz}@oregonstate.edu

ABSTRACT

Throughput-oriented many-core processors demand highly efficient network-on-chip (NoC) architecture for data transferring. Recent advent of silicon interposer, stacked memory and 2.5D integration have further increased data transfer rate. This greatly intensifies traffic bottleneck in the NoC but, at the same time, also brings a significant new opportunity in utilizing wiring resources in the interposer. In this paper, we propose a novel concept called Equivalent Injection Routers (EIRs) which, together with interposer links, transform the few-to-many traffic pattern to many-to-many pattern, thus fundamentally solving the bottleneck problem. We have developed EquiNox as a design example. We utilize N-Queen and Monte Carlo Tree Search (MCTS) methods to help select EIRs by considering comprehensively from topological, architectural and physical aspects. Evaluation results show that, compared with prior work, the proposed EquiNox is able to reduce execution time by 23.5%, energy consumption by 18.9%, and EDP by 32.8%, under similar hardware cost.

1. INTRODUCTION

Throughput-oriented processors (e.g., GPUs) have been increasingly used to speed up a wide range of conventional and emerging applications. Due to the large number of cores in the processors, networks-on-chip (NoCs) have been gaining significant research interest [1, 2, 3, 4, 5, 6, 7, 8] to provide low-latency and high-throughput on-chip communication. Meanwhile, with the recent advent of silicon interposer and 2.5D integration technology, memory chips can be integrated with the processor chip in one package to provide dramatically increased memory throughput [9, 10, 11, 12]. However, this places a huge pressure on the NoC component. Unfortunately, existing schemes have turned out to be ineffective when applied to these interposer-based systems. With more advanced interposer technologies on the horizon [13, 14, 15, 16], the performance gap between the memory and NoC will likely get even widened. Thus, it is imperative to redesign the NoC accordingly to meet the requirements of interposer-based throughput processors.

Silicon interposers bring both major problems and opportunities to throughput-oriented processors. On the one hand, the many-to-few-to-many traffic pattern in those processors may cause a bottleneck in the reply network, where data reply packets that are destined to the many cores are injected through only a few injection nodes. With the boosted memory bandwidth in stacked memory in interposer systems, this injection bottleneck is greatly intensified and is only getting worse with future memory technologies. On the other hand, interposer contains multiple dedicated Redistribution Layers (RDLs), which provides abundant wiring resources that are currently underutilized [14, 17]. These wiring resources have electrical characteristics that are similar to that of on-chip links [18]. This provides a basis for exploiting a hybrid use of on-chip and interposer components.

In this work, we explore the new wiring opportunities brought by the interposer to address the intensified injection bottleneck that is also caused by the interposer. Given the root cause of the injection bottleneck is the few-to-many traffic pattern, we propose *Equivalent Injection Routers (EIRs)* that transform the traffic to many-to-many pattern, thus fundamentally solving the bottleneck problem. This is achieved by providing each injection point with a group of equivalent injection routers, all of which have "equivalent" capability in terms of accepting and distributing the injecting traffic. The required additional interconnects are provided by the redistribution layers in the interposer. While the concept of EIRs is straightforward, selecting the set of equivalent routers requires comprehensive consideration from topological, architectural and physical aspects. This leads to a large design space as explained later in Section 3.

To demonstrate the feasibility of the proposed EIR approach, we have developed *EquiNox* as a design example. The scheme employs a N-Queen based cache bank placement as the basis for selecting equivalent routers. As the solutions of N-Queen are not unique, a scoring policy is developed to select the placement that minimizes network congestion and maximizes EIR potential. The groups of EIRs are then selected by a carefully designed Monte Carlo Tree Search (MCTS) method to search through the design space, while balancing the number of EIRs, their impact on the network, the needed interposer links, the length of the links, and the number of cross-points in the RDLs. The network interface architecture is also enhanced to support the increased injection flexibility from EIRs. The proposed EquiNox is able to meet the requirements of topologically equivalent, architecturally efficient and physically viable EIR designs. Evaluation on a wide range of benchmarks shows that EquiNox achieves 47.7% reduction in execution time and 55.0% reduction in energy-delay product (EDP) compared with a single network scheme, and 23.5% reduction in execution time and 32.8% reduction in EDP compared with a separate network scheme.

The main contributions of this paper are as follows:

- Analyzing new opportunities and challenges of NoCs in silicon interposer-based throughput processors;
- Proposing the concept of Equivalent Injection Routers to provide a novel solution to address NoC bottleneck;
- Developing EquiNox as a case in point to demonstrate the effectiveness of the proposed approach.

The rest of the paper is organized as follows. Section 2 provides more background and motivation on interposer-based throughput processors and the need to address the intensified injection bottleneck. Section 3 proposes the concept of

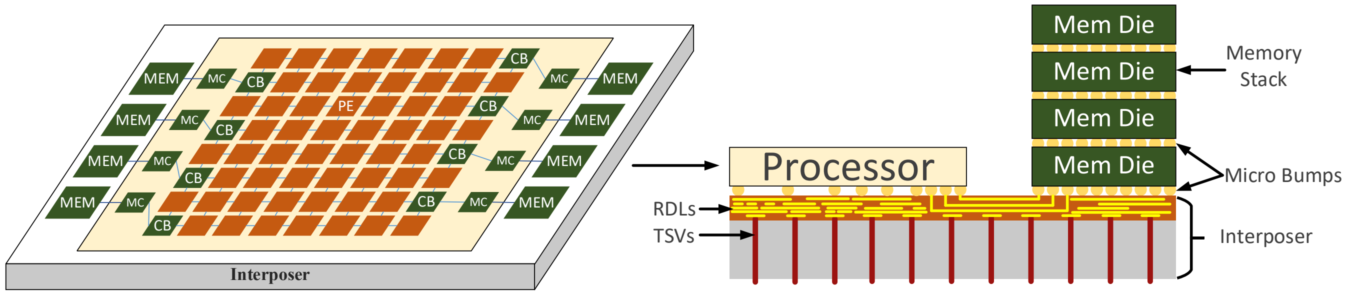


Figure 1: Overview and cross-section view of an interposer-based throughput-oriented processor.

Equivalent Injection Routers and analyzes the challenges to realize it. Section 4 describes the proposed EquiNox in details. Section 5 discusses the evaluation methodology, and Section 6 presents simulation results. Finally, related work is summarized in Section 7, and Section 8 concludes the paper.

2. BACKGROUND AND MOTIVATION

2.1 Interposer-based Throughput Processors

To meet the growing demand of high-bandwidth and low-latency memory in many-core processors, 2.5D integration systems have been proposed and commercially yielded to achieve wafer-level integration of processor dies and memory dies. Figure 1 depicts the architecture of a typical silicon interposer-based throughput processor. The interposer is a silicon substrate that provides mechanical and electrical characteristics to integrate multiple dies [14]. On the left side of Figure 1, the processor die is represented by the large light yellow square. Eight memory dies are located outside the processor die, denoted as the green squares labeled with “MEM". In the processor die, processing elements (PEs) and last level cache banks (LLCBs or simply CBs) are placed in a tile-based fashion. Each cache bank is connected with a dedicated memory controller (MC) that interfaces with a memory die. Each memory die is actually a die stack, which is composed of several individual dies that are vertically placed on top of each other to form a 3D die stack. This technology is known as the High Bandwidth Memory (HBM) [19, 20]. HBMs can greatly benefit from integration in 2.5D systems, since the in-package interconnects offered by the interposer have superior physical properties than off-chip interconnects.

In interposer-based processors, the MCs are usually located near the edge of the processor die to ensure short and fast interposer connections with the memory stacks. As each MC is connected with a CB, only a few CBs exist that are shared by all the PEs. While the locations of the MCs are less flexible, the placement of CBs can be adjusted to achieve more efficient on-chip communication. Different from the conventional many-core CPUs, the PEs in throughput processors (e.g., Stream Multiprocessors (SMs) in GPUs) have little inter-PE communication, but instead communicate with CBs (and then memory stacks) directly. PE-generated request packets are sent to CBs through a *request network*. Reply packets can be generated directly by the CBs in case of cache hits. Otherwise, the CBs will first fetch data from the memory and then generate reply packets containing the data. The reply packets are sent back to the PEs through a *reply network*. This traffic flow from the many PEs to a few CBs and then back to the many PEs is commonly referred to as the Many-to-Few-to-Many (M2F2M) traffic pattern in throughput-oriented processors [2, 21].

Figure 1 also shows the cross-section view of the silicon interposer structure. The processor die and memory stacks are integrated with the interposer via micro-bumps (μbumps) in a face-down fashion (flip-chip packaging technology)[10, 22, 13]. For example, when integrating the processor die with the interposer layer, the die is first flipped, so the surface of the chip is facing down and attached to the interposer. As a result, μbumps consume chip surface area. To connect the processor die with the memory dies, wires are routed through the interposer layer with support of the μbumps. Each wire must have a corresponding μbump to provide the electrical connectivity [10, 13, 14]. Wires in the interposer layer are implemented by fine-pitched metal layers called Redistribution Layers (RDLs) that provide high-bandwidth and low-latency interconnections [11]. To achieve that, the material of the RDLs is usually copper instead of aluminum for lower resistivity and better scalability (higher wire density). However, the Damascene process [23] needs to be used to deal with the poor oxidation and corrosion resistance of copper. In practice, to yield sub-micro pitch metal layers, interposer RDLs may employ a more complex dual-damascene process [24]. Below RDLs, Through-Silicon Vias (TSVs) are used to transfer electrical signals from RDLs to the outside of the package[1].

2.2 Intensified NoC Bottleneck

While the use of silicon interposer brings many benefits, it also worsens the injection bottleneck in the on-chip network. Specifically, each PE node or CB node is associated with an on-chip router, and the routers are connected to form the NoC. A node sends/receives packets to the NoC through a network interface (NI). As mentioned, there are two networks in the NoC: a request network and a reply network. The reply network carries much heavier traffic than the request network. This is because typical workloads for throughput processors have a lot more reads than writes. Read requests are short packets, but read replies are long packets containing cache line data. Our simulation results also confirm this, showing that reply traffic (including read reply and write reply) accounts for 72.7% of the total NoC traffic in terms of bits, and only 27.3% is request traffic (read request and write request). Therefore, the reply network is more susceptible to congestion than the request network [3, 5]. As all these heavy reply traffic is eventually injected into the reply network through a few CB nodes, these injection points (CB nodes) become the performance bottleneck of the entire NoC [2].

In interposer-based systems, the injection bottleneck is

[1]The C4 bumps at the bottom of the interposer layer are not shown in the figure for better clarity.

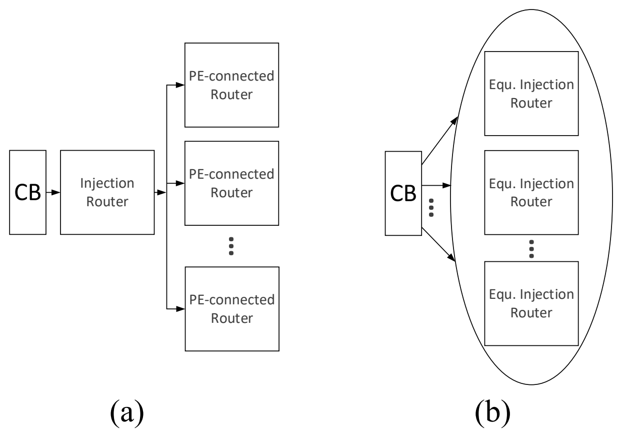


Figure 2: (a) Existing injection for a CB node; (b) Each CB has a group of equivalent injection routers (EIRs).

intensified as the throughput of HBM is significantly higher than that of conventional DRAMs. The second generation of HBM achieves up to 256GB/s [25] which is about 10X higher than GDDR5. This dramatic improvement in memory bandwidth has put a huge traffic pressure on the injection points in the reply network. To-date, only a few of works [2, 4, 5, 6, 7] have targeted the NoC injection bottleneck, but none of them have examined the issue in interposer-based systems where the throughput demand are drastically different. Alternatively, some works exist (e.g., [14, 26]) that utilize interposer resources to improve on-chip networks in many-core CPUs. However, as the M2F2M communication is very different from the all-to-all traffic pattern in CPUs, it is ineffective to adopt these designs for throughput processors. More discussions and quantitative comparisons are provided in later sections, but essentially, without a more specific and effective solution, the gap between the demanded data transfer rate in new memory technologies and the supported rate in current injection points will continue to be widened in the near future.

2.3 Interconnects Opportunity in Interposer

Although the injection bottleneck is worsened by the stacked memory and 2.5D integration, the features of interposer also open up new opportunities for interconnection. First, there is much vacant space under the processor die to route wires in the RDLs. This is because die-to-die interconnects and the associated μbumps are placed near the boundaries of the processor die and memory stack, as shown in Figure 1. This leaves the majority of the area under the processor die unused. Second, RDLs are composed of multiple metal layers. Although the total number of RDLs is limited due to yielding cost, substantial wiring resources are available even with the layers in the current RDLs. Third, the wire latency in the interposer is comparable to that of the die [18]. This allows a hybrid use of interposer links and on-chip links without causing concerns on unmatched signal transfer latency. Moreover, the floor planning of wires in RDLs is independent from that of the processor or memory dies (except for the interfacing μbumps), so the cross-layer wiring complexity within RDLs is not exposed to other system components.

With the above advantages, the next question is how the abundant wiring resources in the interposer can be utilized more efficiently to help with system designs, such as addressing the intensified injection bottleneck.

3. EQUIVALENT INJECTION ROUTERS

3.1 Eliminating Few-to-Many Bottleneck

The root cause for the reply injection bottleneck is a mismatch in quantity, where a *few* injection routers (i.e., CB-connected routers) need to handle all the injection traffic that is destined to the *many* PE-connected routers. Therefore, a fundamental solution to this problem is to somehow transform the “few-to-many" traffic pattern to a “many-to-many" traffic pattern. To this end, we propose the approach of *Equivalent Injection Routers (EIRs)* to eliminate the injection bottleneck. In the existing architecture as shown in Figure 2(a), the injection traffic from a CB is bottlenecked at the CB-connected injection router. In contrast, the proposed approach in Figure 2(b) provides a group of injection routers that have equivalent capability in terms of accepting and distributing the injection traffic from a given CB. Each CB has its own group of EIRs that are located strategically, so injected packets can be quickly distributed. Collectively, all the EIRs create many injection points to realize the many-to-many traffic pattern. However, implementing the idea of EIRs in conventional processors is very difficult, as it requires additional interconnects between a CB and all the EIRs in the group of that CB. This is where the RDLs in the interposer become useful. Since RDLs are underutilized and largely independent from the main NoC, RDLs can be a great resource to route the needed interconnects. In a sense, EIRs provide a nice solution that utilizes the wiring opportunities brought by interposer to address the intensified injection bottleneck that is also caused by interposer (and stacked memory) in the first place.

While the concept of EIRs looks straightforward, selecting the set of equivalent routers requires comprehensive consideration from topological, architectural and physical aspects, as explained below.

3.2 Considerations of Selecting EIRs

3.2.1 Topologically Equivalent

Every EIR in a CB's group is topologically equivalent in the sense that the CB directly connects with every EIR and can use any EIR for packet injection. Therefore, the first consideration is to determine the optimal number of EIRs in a group. At one end of the spectrum, if there is only one EIR per group, the design regresses to the existing architecture in Figure 2(a) where injection is congested. At the other end of the spectrum, if a group includes all the PE-connected routers as the EIRs, the CB can basically send packets to any PE in just one hop through the EIRs. In that case, however, the capacity in transferring traffic out of the CB is way higher than what the CB can possibly inject, thus leaving most of the added links (in the interposer) between the CB and EIRs idle. Therefore, the optimal number of EIRs should be determined carefully.

3.2.2 Architecturally Efficient

Another consideration is whether the selection of EIRs is architecturally efficient, even with the same number of EIRs per group. For example, with 4 EIRs in a group, there are numerous combinations that may have dramatically different architectural efficiency. Figure 3 presents an example where the blue group and the gray group both have 4 EIRs. As the

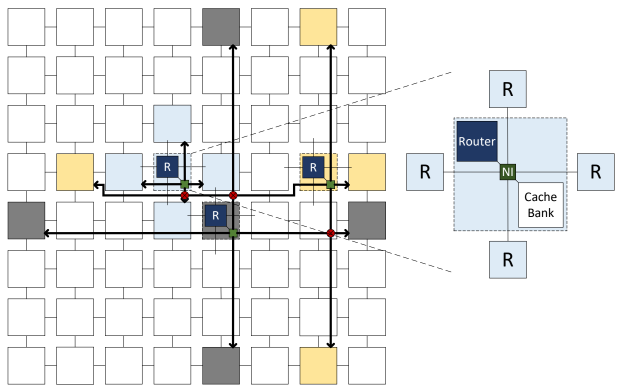


Figure 3: Different examples of a 4-EIR group.

injection routers in the blue group are located closely, the region would easily become a hot zone that leads to high queuing latency for injected packets. In contrast, the gray group distributes the EIRs further into the network, thus lowering the contention among injection traffic and being more architecturally efficient than the blue group. Due to the large number of combinations under a given number of EIRs in a group, it can be difficult to identify the optimal EIR selection in the design space. Note that, while it is beneficial to distribute EIRs across the network, the design in the gray group requires long wires and increases the probability of having wire intersection, both of which place additional constraints on physical viability, as discussed next.

3.2.3 Physically Viable

The positions of EIRs should also be selected in a way that is friendly for physical implementation. There are three main constraints due to the unique characteristics of interposer-based systems: 1) length of interposer links, 2) number of intersection points in redistribution layers (RDLs), and 3) area overhead of μbumps.

First, shorter interposer links are preferred, as long links need repeaters which would require active silicon interposers. Active interposers face greater thermal challenges and complexity than passive interposers. Consequently, EIRs cannot be positioned too far away from the CB.

Second, wire intersection in the interposer needs to be minimized, as intersection points require separate metal layers in the RDLs. For example, in Figure 3, at least two layers are needed to handle the three points of intersection (the three red dots). Due to the dual-damascene process, yielding complexity increases exponentially as more metal layers are included [23, 27]. The process is very costly because of the operations in cleaning residual photoresist and protecting hydrophilic low-κ dielectric films [28].

Third, because of the face-down integration in 2.5D integrated chips, μbumps consume on-chip area of the top dies, e.g., the processor die and memory dies. Every interposer wire needs a dedicated μbump to ensure electrical connectivity with other dies. Taking $40\mu m$ pitch μbumps [22] as an example, each 128-bit bi-directional link consumes around $0.34mm^2$ μbump area. This overhead can be quite substantial when the number of interposer links is large (e.g., prior works on CPU NoCs need hundreds of interposer links). Thus, it is challenging to place EIRs strategically that would require much fewer interposer links, while still being able to avoid the reply injection bottleneck.

3.2.4 Other Complications

In addition to the above three aspects, there are other factors that may potentially affect the design of EIRs. One major issue is the placement of CBs that has considerable impact on system performance. For instance, if multiple CBs are placed closely at the same row, the contention among injection traffic would be very high even with the help of EIRs. Also, the eight nodes surrounding a CB node have more injection traffic, so it is better not to include these surrounding nodes as EIRs which would otherwise draw even more injection traffic. All these factors, compounded by the choice of the number of EIRs in a group, the different combinations of EIRs, and the resulting length/number/intersection of interposer links, make the approach of EIRs challenging (but also interesting). In the next section, we present a design example that integrates N-Queen and Monte Carlo Tree Search methods to explore this large design space and materialize the benefits of EIRs.

4. DESIGN EXAMPLE: EQUINOX

4.1 Overview

In this work, we have developed *EquiNox* as a case in point to demonstrate the feasibility and effectiveness of the proposed approach on using equivalent injection routers (EIRs). EquiNox contains the right combination of several design elements to meet the requirements of topologically equivalent, architecturally efficient and physically viable EIR designs. The scheme employs a N-Queen based cache bank placement as the basis for selecting equivalent routers. Since the solutions of N-Queen are not unique, a scoring policy is developed to select the placement that minimizes network congestion and maximizes EIR potential. The actual group of EIRs is then selected by a carefully designed Monte Carlo Tree Search (MCTS) method. The proposed MCTS balances the number of EIRs and their capability in distributing the injection traffic. It also helps to determine the locations of EIRs by simultaneously reducing the number of needed interposer links, the length of the links, and the number of cross-points. Finally, a few low-cost but critical changes are applied to the NI architecture to support the increased injection flexibility that is offered by multiple equivalent routers. The following subsections describe these design elements in more detail.

4.2 Contention-aware CB Placement

As the basis of the EIR approach, we first present a last-level cache-bank placement that is benign to equivalent injection routers.

Hints from Existing Placements: Several popular CB placements exist, such as Top, Side, Diagonal and Diamond [21] that are originally proposed for the all-to-all traffic pattern in many-core CPUs. We analyze these placement schemes to obtain some hints for finding placements that are good for the traffic patterns in throughput processors. To get a visual intuition of the traffic situation under different placements, Figure 4 draws the heat map of the average number of cycles that a flit experiences when traveling through a router in the reply network (where the injection traffic forms the few-to-many traffic pattern). Each colored block denotes a router. A brighter block means that flits spend more cycles in this router. The scale is shown on the right, and the variance of cycles among the routers is shown below each sub-figure. For the Top and Side placements, it can be seen that there are severe

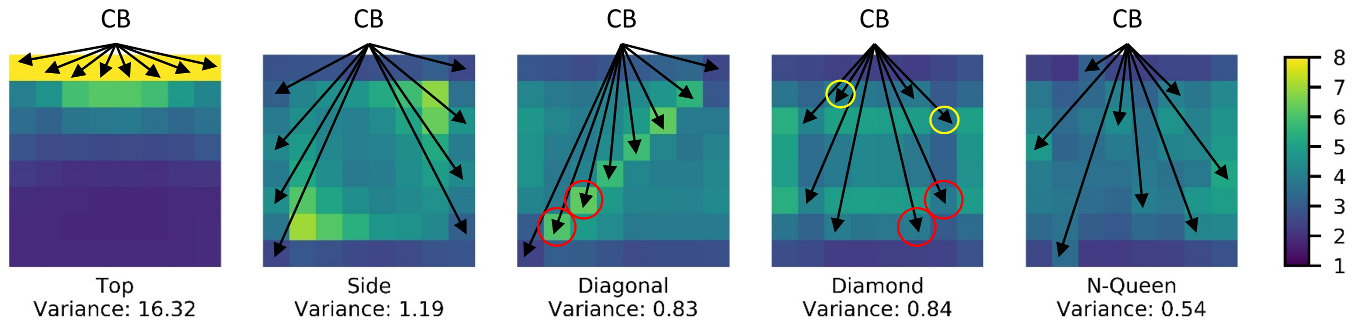


Figure 4: Heat map of average router travel through cycles in different CB placement.

delays that stress CB nodes and/or the nodes surrounding them. This is due to the higher probability that reply packets may encounter each other when CBs are placed at the same row or the same column.

For the Diagonal or Diamond placements, since there are no CBs at the same row/column, the traffic is more balanced, as indicated from the significantly reduced variance values. However, these two placements may cause intersection problems for EIRs due to diagonally neighboring CBs. Consider the two adjacent CB nodes such as the two red-circled nodes in the Diamond (or Diagonal) placement. If the upper CB node has a horizontal (e.g., x-) interposer link connecting an EIR and the lower CB has a vertical (e.g., y+) interposer link connecting another EIR, the two links would inevitably intersect with each other even if both links are only one-hop long. This increases the number of RDLs and yielding cost. In comparison, for the two yellow-circled nodes that are not directly positioned diagonally, intersection can be avoided if the lengths of interposer links are selected carefully. Moreover, neighboring diagonal CBs also increase the contention of injection traffic, which could be mitigated if they are not positioned diagonally.

***N*-Queen Based Placement:** The above analysis prompts us to find a CB placement that minimizes the alignment of CBs in the same row, column or diagonal, so as to reduce traffic contention and be friendly with interposer wiring. These considerations lead us to utilize the N-Queen algorithm that is originally proposed to place N queens on a chessboard where no two queens can capture each other. If used for placing the CBs, such placement ensures that there is only one CB node in a row or a column and there are no CB nodes in diagonal (whether neighboring or not). An example is shown in Figure 4 where the placement is very effective, with a variance of only 0.54. This is 35.7% lower than the Diamond placement and 96.7% lower than the Top placement. Moreover, the fact that only one CB node is on any diagonal in the N-Queen placement also decreases the probability of having wire intersection. This increases the flexibility of selecting equivalent injection routers.

Scoring Policy: The N-Queen algorithm does not generate a unique solution. Many N-Queen placements exist, and they may have different impact on traffic congestion. To assess different placements quantitatively, we introduce a concept of *hot zone*. The hot zone of a CB node is defined as the 8 nodes that surround the CB node, as illustrated in Figure 5. In particular, the 4 nodes that are connected directly with a CB node are called *Direct Access Zones (DAZs)*. DAZs are very congested as any injected packet is forwarded through DAZs as the first hop. The other 4 nodes at the four corners are called *Corner Access Zones (CAZs)*. Packets have a high probability of being forwarded to CAZs as the second hop. If there is an overlap between the DAZ hot zone of a CB node with the CAZ hot zone another CB node, traffic congestion is greatly exacerbated. Therefore, hot zone overlaps can be used as a metric to assess the quality of N-Queen placements. It is worth pointing out that, in N-Queen placement, it is not possible to have DAZ-DAZ or CAZ-CAZ overlaps, which is another reason why N-Queen is a good placement strategy.

The following scoring policy is introduced. Under a given placement, we calculate a penalty score for each node (tile) in the network, and the summation of the penalty scores of all the nodes is the final penalty score of that placement. Among the four direct neighbors of a node, if m of them are hot zone overlaps, the penalty score of this node is $\sum_1^m$. We use this policy rather than simply adding m points, to reflect the compounded delay by multiple hot zone overlaps. For example, in Figure 5, to calculate the penalty score of the red node, we notice that two of its direct neighbors (light yellow nodes) are hot zone overlaps, so the penalty score of the red node is 1+2=3. Note that the node above the red node is a DAZ but is not a hot zone overlap; whereas the two yellow nodes are DAZ-CAZ overlaps. In case of an 8×8 network, there are 92 different N-Queen placements. The one with the lowest score is chosen, as shown in Figure 5. For larger networks, a similar procedure is followed to generate a number of N-Queen placements, and the least penalized one is selected.

4.3 Selecting EIRs with MCTS

After the CB placement is decided by the above N-Queen and scoring policy, the next step is to select the EIRs. Two observations can be used to simplify the complexity of the selection. First, for a given CB, it is better to distribute EIRs on different directions from the CB, as two EIRs on the same direction would cause contention in that direction. Second, it is better to place EIRs within a few hops from a CB to avoid using long interposer wires and to reduce intersection. However, the design space after these simplifications is still quite large. In the example of an 8×8 network, there are 1.7×10^{10} possible combinations of EIR selections, even if we limit EIRs to be within 3 hops of the corresponding CB node. Therefore, a systematic search approach is needed to identify a good EIR selection. In this work, we develop a search method based on Monte Carlo Tree Search (MCTS).

MCTS is a classic search algorithm in machine learning [29] and has been recently employed to enhance the AI algorithms in the game of Go, Shogi and StarCraft II [30, 31, 32] to search their huge solution spaces. We adopt MCTS

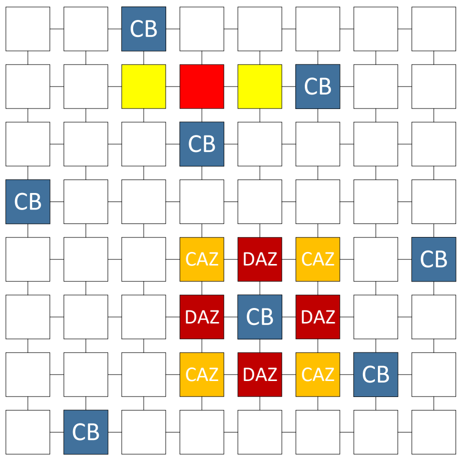


Figure 5: N-Queen placement.

due to the inherent similarity between placing EIRs in our problem and placing stones in the game of Go. Other search algorithms might work, such as genetic algorithm (GA) or simulated annealing (SA), but likely at a lower efficiency due to the additional mathematical transformation and less effective problem representation. For example, to utilize GA, a natural representation is to use a 64-bit gene for an 8×8 network, where each bit is either 0 or 1 to indicate if that node is an EIR. This unnecessarily expands the problem space to 2^{64} (or 1.8×10^{19}) and introduces numerous invalid solutions during crossover and mutation operations. Similar issue on problem formulation exists in SA as well.

Our proposed search method follows a typical MCTS, with a few customization and optimizations specific to EIR selection. The search process builds a search tree iteratively. Each iteration consists of four steps, namely selection, expansion, simulation and backpropagation, as shown in Figure 6.

(1) *Selection*: Search starts from the root node (e.g., an empty state with no EIR selected) and recursively selects a child node until reaching a current leaf node (e.g., a few EIRs added from previous iterations). This step selects a promising path from the current search tree, so the path can be expanded in the following steps. The selection is based on an Upper Confidence Bound (UCB) formula[2] that balances exploitation, which maximizes the rewards from known nodes, and exploration, which explores unknown nodes for potentially higher rewards [33].

(2) *Expansion*: Assuming the above leaf node is not a terminating node (e.g., last EIR added), this step randomly chooses a possible successive state (e.g., add another EIR or a group of EIRs), attaches to the leaf node, and expands the tree by one level.

(3) *Simulation*: Possible outcomes following the expansion are "simulated" by performing a rollout policy. In case of the Go game, this step means that multiple moves are performed speculatively if the current move is indeed made. Similarly, additional EIRs are selected speculatively (but not actually selected).

(4) *Backpropagation*: An evaluation function estimates the value of the rollout based on a set of defined rules and grading policy. The evaluation score is then backpropagated to all

[2]Defined as $v_i + C \times \sqrt{lnN/n_i}$, where v_i is the estimated value of the chosen child, n_i is its total number of visits, and N is the total number of visits of its parent node. C is a balancing parameter.

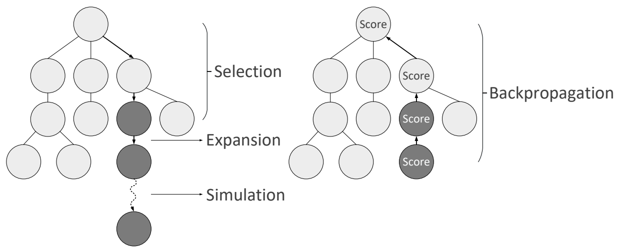


Figure 6: Four primary stages of MCTS.

the nodes on the path, starting from the expanded node to the root node. The score is accumulated to the existing scores of those nodes.

At the end of the first iteration, the level-1 child node of the root with the highest accumulative score is considered as near-optimal, and the corresponding EIR selection represented by that node is added to the final EIR selection. This information is carried over to the second iteration as part of the new root state. The second iteration finalizes the EIR selection of a level-2 child (with the highest accumulative score) of the root, and so on so forth, until the EIRs of all the CBs are selected.

The above search process adds EIR one by one. While this is correct, during implementation we observe that the search tree can be as deep as 24 levels for an 8×8 network and even deeper for larger networks. This reduces search efficiency significantly. To address this issue, instead of adding EIR one at a time, we add EIRs group by group. Specifically, each node expansion adds the selection of all the EIRs belonging to a CB node. Therefore, the tree depth equals exactly the number of CB nodes, which is usually limited in a processor.

An important component of MCTS is the evaluation function in backpropagation. We integrate four metrics in the evaluation function to reflect the considerations on EIR efficacy and physical viability: (minimizing) the max of EIR traffic load, average hop count, number of intersection points, and link length. The first metric estimates the total amount of traffic that each EIR needs to handle for all the PE nodes, and the objective is to minimize the maximum one across all the EIRs. This metric helps to balance traffic load among EIRs to avoid hotspots. This is particularly useful as some CB nodes may have fewer EIRs, e.g., due to the consideration of avoiding intersection or simply due to boundary constraints. The second metric uses average hop count to approximate packet latency. The third and fourth metrics consider the physical constraints of RDLs in the interposer. All the four metrics can be easily calculated under a specific EIR selection and CB placement, assuming each PE has relatively similar traffic load. Owing to this, metric calculations can be performed quickly in each backpropagation step to guide the search process; whereas detailed full system simulations with actual workloads are conducted later only for EIR selections that are found to be promising by MCTS. The evaluation function sums the four metrics (after normalization). Lower function values indicate better EIR selections.

We implement the above MCTS model in Python 3.7.3 [34] and execute on an x86_64 Linux server, equipped with an Intel E5-2630v3 32-core processor and 64GB memory. The search process turns out to be quite efficient. For instance, for an 8×8 network, MCTS can stabilize to a near-optimal EIR selection in less than 10 hours by assessing only 0.047‰ of the entire solution space. Figure 7 shows the best design found by MCTS in this case. EIRs with the same color

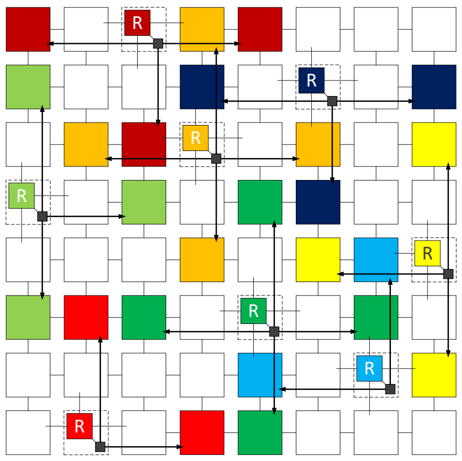


Figure 7: EquiNox in case of an 8x8 NoC. Blocks with the same color belong to the same CB group.

belong to the same group of a CB (in MCTS, we do not allow an EIR to be shared with more than one CB). While the search process is carried out without human intervention, it is interesting to see that MCTS seems to be able to "synthesize" several design attributes of good EIR selections. First, as can be seen, all the EIRs are placed exactly 2 hops away from each CB, despite the up to 3-hop flexibility in the constraint. Closer examination verifies that 2-hop away EIRs bypass both DAZ and CAZ hot zones surrounding the corresponding CB node. Second, intersection is completely avoided, thus requiring minimally only one RDL in the silicon interposer. Furthermore, interposer links with 2-hop length can be fit into one clock cycle, thus avoiding the use of repeaters and active interposers. Note that those 2-hop links are routed in the interposer, so they do not interfere with the placement of regular links in the processor die.

As the network size increases, one might think that EIRs located multiple hops away from CBs are more efficient. However, the 2-hop away EIRs actually work very well for larger networks. First, the number of routers to distribute the injected packets increases geometrically as the packets are forwarded further into a network (e.g., 4 routers after the first hop, 16 routers after the second hop, etc.). Thus, for larger networks, the bottleneck is still at the region close to injection points. Second, we have observed that, the contention delay from injection quickly drops after one or two hops after the injection points, regardless of the network sizes. Therefore, using interposer links to bypass the first two hops are sufficient to avoid injection contention even in large networks. For extremely large networks, if the need arises, it is possible to use slightly longer interposer links (e.g., 3-hop links). Intersection can still be avoided as those networks provide more space to place non-intersected links. However, as we show later in the Evaluation section, two hops are more than enough even for 16×16.

4.4 Modifications to Microarchitecture

To implement EquiNox, some modifications are needed to the architecture. In the original NI structure, an injected packet from the last-level cache bank is serialized by the NI core logic and then stored in a packet injection buffer. This buffer directly connects to the local CB-connected router. The size of the buffer is usually a few packets, although only one flit is sent to the connected router in a given cycle. With the use of EIRs, the CB-connected NI needs to connect to multiple equivalent injection routers. Consequently, the NI structure needs to be modified, as depicted in Figure 8. The main change is that the injection buffer is split to five single-packet buffers, where four of them are connected to the four EIRs that are two hops away through interposer, and the remaining one is connected to the original local router. To reduce design and verification effort, all the CB-connected NIs use this architecture, even though some ports can be left idle if there are fewer EIRs connected. Later in the evaluation section, we configure the original NI to have one packet-sized injection buffer and the modified NI to have five packet-sized injection buffers to appropriately account for the overhead. In addition, a de-multiplexer is inserted to switch injected packets to different injection buffers, and a buffer selection signal is generated by the *Buffer Selector* when the NI core logic is processing a packet.

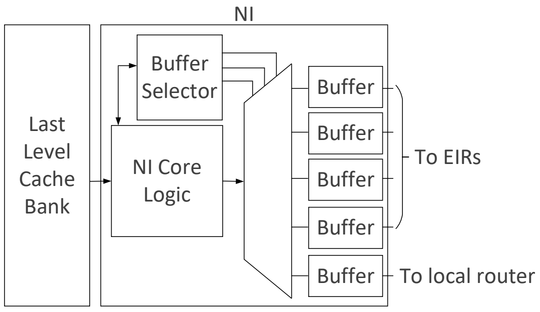


Figure 8: CB-connected NI architecture in EquiNox.

To avoid detouring, injected packets are only allowed to use the EIRs on the shortest paths (or the local router which is also on the shortest path), even though other EIRs might have less traffic. This is fulfilled by the Buffer Selector. Specifically, the relative position of a packet's destination (x_d, y_d) with regard to its source (x_s, y_s) is first generated. Based on the relative position $(\Delta x, \Delta y)$, there are 8 possible relative directions, of which 4 are right on the axis (either Δx or Δy is 0) and 4 are inside the four quadrants. If a destination is on an axis, there is one and only one EIR that is on the shortest path, and that EIR is selected. If the destination is inside one of the four quadrants, there are up to two shortest-path EIRs exist (except for boundary cases). In this scenario, round-robin is used to select the EIR. In either case, if the buffer that connects to the to-be selected EIR(s) is not available, the packet is injected into the local CB-connected router. Detailed selection policy is presented in Buffer Selection 1. Note that it is not possible for both Δx and Δy to be 0 as MC nodes do not send packets to themselves.

While not shown in Figure 8, the EIR on the receiving side needs to accept injected packets from the split NI injection buffer. Therefore, one input port is added to the EIR. Note that this is needed only for routers that are selected as EIRs and only for the reply network, while routers in the request network is unchanged. Some EIRs are located on the boundary, so an unused input port might already be available, if the boundary routers use the same router template as the non-boundary ones (to reduce verification cost). Either way, the amortized overhead is only a few percent and much better than existing alternatives to mitigate the injection bottleneck, as presented in Section 6.

Deadlock freedom is a related critical issue that should

Buffer Selection 1: Buffer Decision Policy

```
if $\Delta x == 0$ or $\Delta y == 0$ then
    if the buffer is available then
        Inject to the buffer associated with the node on
          destination direction
    else if local buffer is available then
        Inject to the local buffer
    else
        Retry next cycle
else if $\Delta x\ != 0$ and $\Delta y\ != 0$ then
    if 2 buffers are available then
        Choose 1 buffer in round-robin fashion
    else if 1 buffer is available then
        Inject the packet to the buffer
    else if local buffer is available then
        Inject to the local buffer
    else
        Retry next cycle
else
    Error
```

be discussed. First of all, EIRs do not affect existing virtual channel (VC) allocation. Packets that are injected from an NI to an EIR use the same VC allocation policy as if the packets are injected to the local router. Also, without detouring, the addition of EIRs do not change the routing policy in the original network. Therefore, if the channel dependence graph of the original network is acyclic, the EIRs do not introduce new cycles and are free from routing-induced deadlocks. Regarding protocol-introduced deadlock, since the request and reply packets are routed through two physical networks, there is no dependence at the endpoints and no protocol-introduced deadlock either. Thus, EquiNox is deadlock-free.

To summarize, with the combination of a N-Queen based CB placement, a set of carefully placed equivalent injection routers determined by MCTS, and the needed microarchitecture modifications to enable correct and deadlock-free operations, the proposed Equnix effectively removes the injection bottleneck and utilizes interposer wiring resources. In the following sections, EquiNox is evaluated quantitatively.

5. EVALUATION METHODOLOGY

The proposed EquiNox design is evaluated using a combination of architecture and RTL level simulators. Due to the lack of a comprehensive cycle-accurate simulator that models HBM and interposer, we have developed an integrated simulation environment by combining and heavily modifying BookSim 2.0 [35], GPGPUSim 3.2.3 [1] and Ramulator [36]. The NoC simulation is performed by the cycle-accurate simulator BookSim 2.0 that includes all the on-chip network resources (e.g., links, routers, network interfaces). We integrate Ramulator with GPGPUSim to enable HBM simulation. The simulated HBM contains 8 chips, each having 4 stacks. There are 64 TSV IOs per channel and 16 channels per chip, totaling 1024 IOs for each chip. The physical layer PHY that has 8 channels and locates on top of each memory stack [37] is also simulated as the interface between HBM and memory controllers. The area and power consumption of NoCs are based on DSENT [38] which is extensible to simulate novel components of NoCs. Following the methodology of prior works on interposer [14, 39, 40], we modify and extend DSENT to model interposer links. To evaluate area overhead more accurately, we use Verilog HDL to implement the RTL of new components in EquiNox that are not modeled in DSENT. A standard VLSI design flow is followed that employs Design Compiler for logic synthesis using 28*nm* process technology [41]. Table 1 lists the key configuration parameters. A 8×8 Mesh NoC is assumed for the main simulations, and 12×12 and 16×16 NoCs are also simulated for scalability study. A wide range of 29 benchmarks from Rodinia [42] and Nvidia CUDA SDK [43] are executed to evaluate the performance of different proposed schemes.

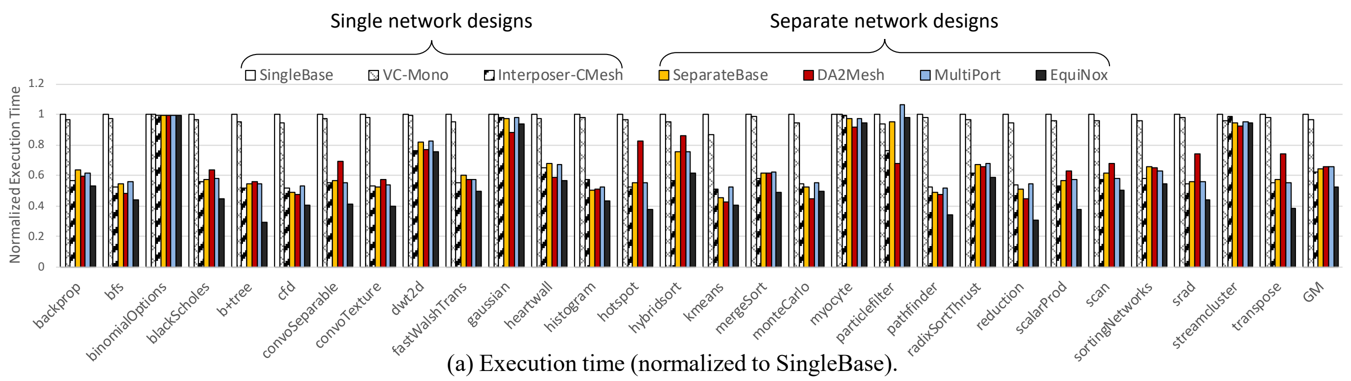
Table 1: Key Parameters in Simulation.

Parameter	Value
Network size	8x8, 12x12, 16x16
Network routing	Minimum adaptive
Virtual channel	2/port, 1 pkt/VC
Allocator	Separable input first
PE frequency	1126MHz
Shared memory / PE	48KB
L1 cache / PE	16KB
L2 cache (LLC) per bank	2MB
# of LLC banks	8
HBM bandwidth	256GB/s per stack
# of Memory dies / stack	4
Memory controllers	8, FR-FCFS

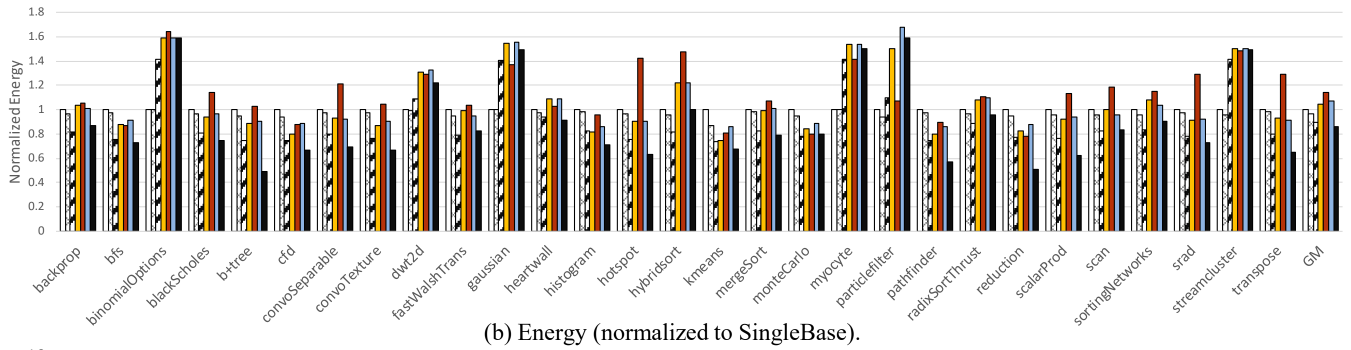
We compare the following seven schemes. In particular, schemes (1), (2) and (3) are based on the single network type where the request and reply networks share the same physical network. Schemes (4), (5), (6) and (7) are based on the separate network type where the request and reply networks have separate physical networks. In both types, we include a baseline, one or two recent but conventional (no interposer) schemes, and an interposer-based design. Below are the details.
(1) SingleBase: baseline for the single network type with Diamond placement and minimal adaptive routing.
(2) VC-Mono [4]: a recent scheme that increases network throughput by allocating all the VCs (monopolization) to either request or reply traffic if only one of them is present.
(3) Interposer-CMesh [14]: a state-of-the-art scheme for many-core CPU NoCs that introduces an additional CMesh network whose links are in the interposer.
(4) SeparateBase: baseline for the separate network type with Diamond CB placement and minimal adaptive routing.
(5) DA2Mesh [5]: a recent scheme that splits the reply network into eight narrow subnets with 1/8 flit size; the network frequency is set to 2.5X of the baseline in [5].
(6) MultiPort [2]: a scheme that uses multiple injection (and ejections) ports for all CB-connected routers to mitigate the reply injection bottleneck.
(7) EquiNox: the proposed scheme with N-Queen placement and MCTS-selected EIRs, as described in Section 4.

6. RESULTS AND ANALYSIS

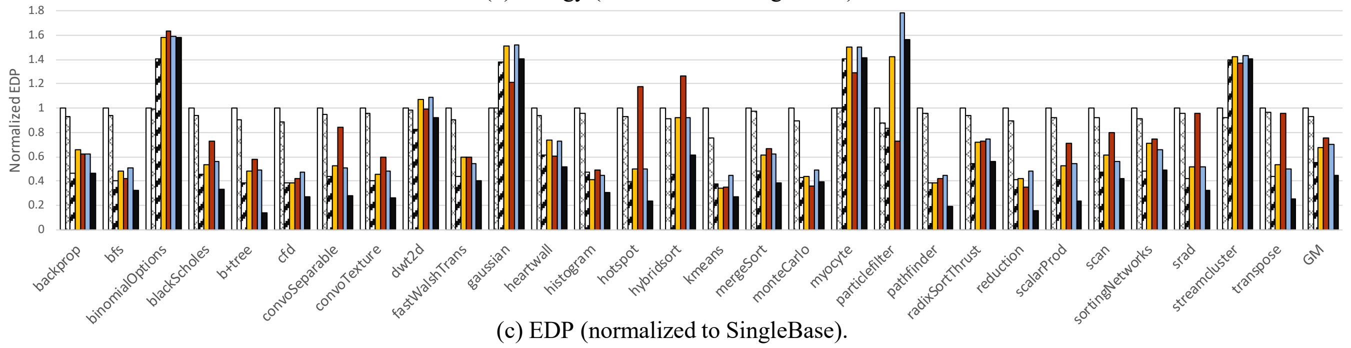
This section presents the detailed evaluation results. As single network schemes and separate network schemes have very different performance-energy characteristics, we thereby juxtapose execution time and energy consumption with ener-



(a) Execution time (normalized to SingleBase).



(b) Energy (normalized to SingleBase).



(c) EDP (normalized to SingleBase).

Figure 9: Comparison of Execution time, energy and energy-delay product (EDP).

gy-delay product (EDP) to reflect the trade-off more accurately.

6.1 Effect on Performance

Figure 8(a) plots the execution time for the seven schemes, normalized to SingleBase. VC-Mono reduces the execution time by 3.6% on average, although large reduction is observed in some benchmarks (e.g., 13.1% in kmeans) because of better utilization of virtual channels via monopolization. However, these two conventional single network schemes (SingleBase and VC-Mono) do not provide additional bandwidth to mitigate the reply injection bottleneck and thus have lower performance than other schemes. The third single network scheme, Interposer-CMesh, achieves 37.9% execution time reduction compared with SingleBase. This is mainly due to the use of an extra network formed by interposer links in the RDLs. The separate network schemes perform noticeably better than the single network schemes in general, as the separate network schemes provide a dedicated network for the reply traffic, thus having more injection bandwidth. DA2Mesh obtains sizable reduction in execution time for heartwall, kmeans, monteCarlo and particlefilter, whereas MultiPort has larger improvement for fastWalshTrans, scan and sortingNetworks. However, the two schemes do not seem to perform much better than the SeparateBase when averaged over all the benchmarks. For DA2Mesh, this is due to the high packet serialization latency from its narrow subnetwork links[3]. For MultiPort, although multiple ports widens the injection bandwidth at CB-connected routers, the injected traffic cannot be quickly transferred out, due to the traffic contention in the hot zone surrounding CB nodes. It is interesting to see that the three conventional separate network schemes (SeparateBase, DA2Mesh and MultiPort) have slightly lower performance than Interposer-CMesh. This shows that it is beneficial to exploit interposer links. Finally, the proposed EquiNox reduces execution time by 47.7% compared with SingleBase and by 23.5% compared with SeperateBase, which is the largest reduction among all the schemes. In particular, EquiNox performs much better than Interposer-CMesh as EquiNox specifically addresses the injection bottleneck that

[3]The improvement of DA2Mesh is smaller here than what is reported in [5] due to differences in benchmarks and settings. As verification, our implemented DA2Mesh performs similarly as in the original paper if those factors are the same.

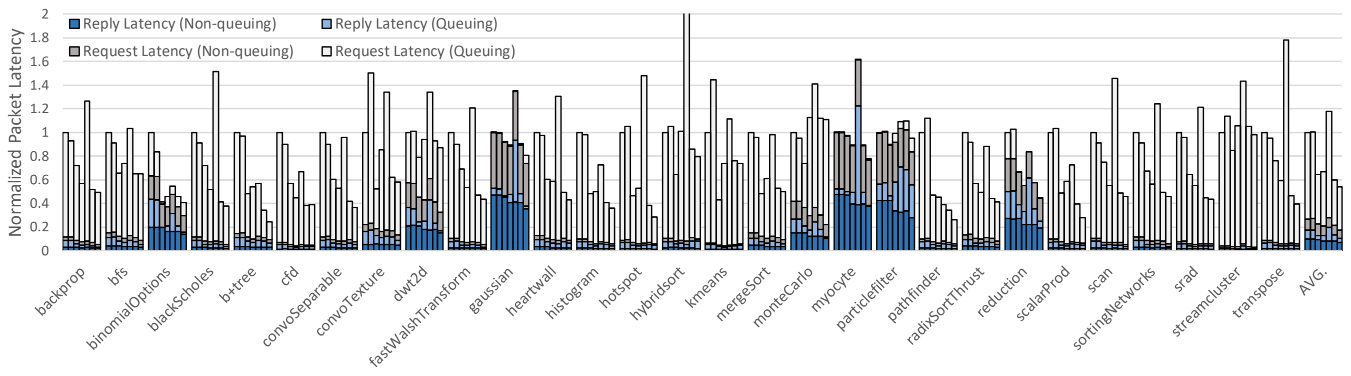


Figure 10: Normalized packet Latency (the bars from left to right are: *SingleBase*, *VC-Mono*, *Interposer-CMesh*, *SeparateBase*, *DA2Mesh*, *MultiPort*, and *EquiNox*).

is unique to throughput processors.

6.2 Effect on Energy

Figure 8(b) compares NoC energy consumption. As can be seen, the three conventional separate network schemes (SeparateBase, DA2Mesh and MultiPort) have the highest energy due to the overhead of two physical networks. While being a separate network scheme, EquiNox has low energy consumption because of small power overhead and large reduction in execution time. On average, EquiNox achieves 15.0% less NoC energy than SingleBase and 18.9% less than SeparateBase.

6.3 Effect on Energy-Delay Product

Figure 8(c) examines the normalized EDP of the compared schemes. The conventional single network schemes (SingleBase and VC-Mono) have higher EDP than the separate network schemes, as single network schemes have higher execution time but similar energy consumption. However, Interposer-CMesh that exploits the interposer layer is able to reduce the EDP by 44.7% compared with SingleBase, and achieves 26.4% and 21.0% less EDP when compared with DA2Mesh and MultiPort, respectively. The results indicate that the interposer-based designs are useful in exploiting the interposer opportunity to achieve better performance-energy trade-off. The proposed EquiNox is able to reduce EDP by 55.0% when compared with SingleBase and by 32.8% compared with SeparateBase. These large improvement further supports the observation that interposer-based designs are promising for high-performance and energy-efficiency. In addition, the EDP of EquiNox is 18.2% lower than Interposer-CMesh, if compared relatively. This demonstrates the need for optimizing the use of interpose links for specific traffic behaviors in throughput processors.

6.4 Reduction of Packet Latency

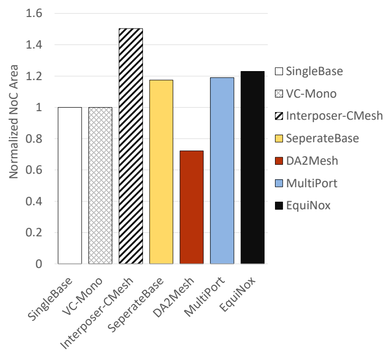
To gain more insights on why EquiNox is able to achieve a large performance improvement, Figure 10 plots the NoC packet latency, breaking down into queuing and non-queuing parts for both request and reply latency. The scheme bars follow the same order as Figure 9 from left to right. Because DA2Mesh has a different NoC frequency, all the results are converted to nanosecond (*ns*) and then normalized to SingleBase to provide an accurate and fair comparison. Due to normalization, for benchmarks such as gaussian and myocyte, the bars do not indicate that their non-queuing latency is high; they just mean that most of the latency is from the non-queuing part.

On average, the figure shows that request latency is considerably higher than reply latency. This might be non-intuitive since the bottleneck is at the reply injection point. However, this is correct because the congestion at reply injection creates a backpressure that is propagated to the request network. Analogous to the classic parking lot problem with a congested exit point, the cars that are the farthest from the exit experience the longest waiting time. Similarly, packets in the request network experience longer NoC latency even though the actual congestion occurs in the reply injection. This trend is consistent with prior observations [2, 4, 5, 7].

Overall, the single network schemes have relatively higher packet latency. This is expected due to the traffic contention between mixed request and heavy reply traffic. With interposer links, Interposer-CMesh reduces packet latency by 35.6%. SeparateBase and MultiPort have similar reduction in packet latency, with 33.1% and 40.2% on average, respectively. The highest average packet latency is observed in DA2Mesh. Further inspection shows that this is mainly caused by a much higher serialization latency. For the proposed EquiNox, it has the lowest reply packet latency as the use of EIRs addresses the bottleneck at the reply injection. It can be seen that the queuing part of the request latency is reduced significantly in EquiNox, due to the backpressure explained above. Compared with SingleBase, EquiNox greatly reduces the request, reply and total packet latency by 44.6%, 40.6% and 45.8%, respectively.

6.5 NoC Area

We have also assessed the area cost of various schemes, plotted in Figure 11. As expected, conventional single network schemes have lower area than separate network schemes. The only exception is Interposer-CMesh, which has an extra concentrated mesh network with 16 routers (links are in the interposer but not the routers). Moreover, these routers have 2x more ports than a basic router because of the need to handle both concentrated traffic and inherent CMesh traffic. This contributes to a higher NoC area for Interposer-CMesh. On the separate network schemes side, DA2Mesh has lower area due to the narrower and simpler routers, whereas MultiPort and EquiNox have higher area than SeparateBase due to the use of extra ports. In particular, with the additional components such as added buffers in NIs and added input ports in EIRs, the proposed EquiNox consumes 4.6% more die area than SeparateBase.



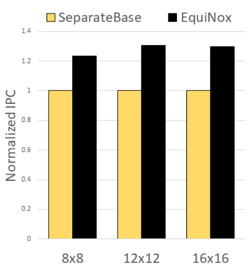


Figure 11: NoC area comparison. Figure 12: Scalability.

6.6 Comparison of μbumps Area

This subsection compares the μbumps area for Inter-poser-CMesh and EquiNox. To support the additional concentrated mesh network, Interposer-CMesh needs 128 uni-directional links between the processor die and interposer, with 256 bits for each link. This leads to a total of 32,768 μbumps for physical and electrical connectivity in Interposer-CMesh. In contrast, the proposed EquiNox has 24 uni-directional 128-bit links and two μbumps for each wire (from the process die to interposer and back to the processor die), resulting in 6,144 μbumps. The μbumps overhead is significantly reduced in EquiNox by 81.25%. The large saving comes from the fact that EquiNox utilizes the M2F2M traffic pattern and strategically places only a few EIRs. Note that neither schemes has intersection of interposer links, so one RDL is sufficient in both schemes.

6.7 Scalability

As analyzed in Section 4.3, the proposed scheme is expected to work well with larger network sizes. To verify this, we follow the same design flow of N-Queen and MCTS for 8×8 to generate EquiNox versions for 12×12 and 16×16 networks. As compared in Figure 12, the performance improvement (average IPC) is 1.31x in 12×12, and 1.30x in 16×16, both of which are greater than the 1.23x in 8×8. Larger networks may have more serious injection bottleneck issue, so the impact of EquiNox becomes greater. These results demonstrate the excellent scalability of EquiNox.

6.8 Discussion

One potential issue is how to deal with the number of CBs that is greater than N. Apparently, there will be more than one CBs on at least one row, column or diagonal. Due to space constraint, we state here without providing proof that, if the number of CBs is greater than N in a $N * N$ layout, placing CBs following the knight-move shape in chess can lead to the lowest occurrence where two CBs are on the same row, column or diagonal. The scoring policy is still applicable except that hot zone overlaps may be between DAZ-DAZ and CAZ-CAZ. The remaining steps are the same as Section 4.2.

If the number of CBs is less than N, the redundant CBs from the N-Queen solution can be randomly deleted, and the scoring policy can be used to select the (near) optimal one.

7. RELATED WORK

Bakhoda *et al.* observe the M2F2M traffic pattern and propose a scheme specifically for this pattern [2]. Some works propose to split a reply network into several reply subnetworks physically [5, 7] or even in time division fashion [6] to gain a higher injection rate for the reply network. Jang *et al.* propose VC-Monopolization design for single network NoC system in GPGPUs, which improves VC utilization and network throughput. These works have been compared in Section 6. The design space in interposer-based 2.5D system is exploited for NoCs to build energy-efficent NoCs for many-core CPUs [14, 26]. In [14], some general cases of designing energy-efficient NoC are explored. In [26], an efficient design of NoCs in chiplet interposer-based CMP systems is proposed. However, those designs have limited effectiveness when adopted directly to interposer-based throughput processors due to significantly different traffic patterns.

Express links have been proposed for both off-chip (e.g., Express Cube [44]) and on-chip networks (e.g., Flattened Butterfly [45] and MECS [46]). Those topologies typically use an extensive number of express links, which are redundant in throughput processors due to minimal inter-PE traffic.

Another approach to mitigate the impact of the injection bottleneck is to compress reply packets to smaller packets, and unzip after after they are injected [47]. Also, it is possible to alleviate injection bottleneck by employing near-data processing in interposer-based systems [39]. With near memory computing nodes in memory stacks, the reply traffic that needs to travel through the reply network is reduced. These two schemes are largely orthogonal and complementary to the our proposed EquiNox. Additionally, Ziabari *et al.* propose an asymmetric network design in the request and reply network where unnecessary links are removed to exploit memory traffic characteristics in GPUs [3]. Zhao *et al.* design a Heterogeneous Ring-Chain network (HRCnet) for the reply network which provides lower area and power consumption and reduces packet conflicts with a ring-based topology [8]. While these works are effective in their targeted contexts, they do not consider interpose-related characteristics.

8. CONCLUSION

Current and future interposer-based throughput processors require high-throughput NoC to support dramatically improved memory bandwidth. However, the wiring resources in the interposer layer have been largely neglected in the past for throughput processors. In this work, we propose a novel concept of Equivalent Injection Routers that solve the injection bottleneck by transforming the traffic pattern from few-to-many to many-to-many. A specific design example, EquiNox, is also proposed to demonstrate how the EIRs can be selected effectively by considering factors from topology, architecture and physical implementation. Evaluation results show significant improvement of EquiNox over prior schemes in both single and separate network designs.

ACKNOWLEDGMENT: We sincerely thank the reviewers for their helpful comments and suggestions. This research was supported, in part, by the National Science Foundation (NSF) grants #1566637 and #1750047.

REFERENCES

[1] A. Bakhoda, G. L. Yuan, W. W. Fung, H. Wong, and T. M. Aamodt, "Analyzing cuda workloads using a detailed gpu simulator," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS'09)*, pp. 163–174, 2009.

[2] A. Bakhoda, J. Kim, and T. M. Aamodt, "Throughput-effective on-chip networks for manycore accelerators," in *Proceedings of the 43rd annual IEEE/ACM International Symposium on Microarchitecture (MICRO'10)*, pp. 421–432, 2010.

[3] A. K. Ziabari, J. L. Abellán, Y. Ma, A. Joshi, and D. Kaeli, "Asymmetric noc architectures for gpu systems," in *9th IEEE/ACM International Symposium on Networks on Chip (NoCS'15)*, p. 25, 2015.

[4] H. Jang, J. Kim, P. Gratz, K. H. Yum, and E. J. Kim, "Bandwidth-efficient on-chip interconnect designs for gpgpus," in *Proceedings of the 52nd Annual Design Automation Conference*, p. 9, ACM, 2015.

[5] H. Kim, J. Kim, W. Seo, Y. Cho, and S. Ryu, "Providing cost-effective on-chip network bandwidth in gpgpus," in *30th IEEE International Conference on Computer Design (ICCD'12)*, pp. 407–412, 2012.

[6] V. Y. Raparti and S. Pasricha, "Memory-aware circuit overlay nocs for latency optimized gpgpu architectures," in *17th IEEE International Symposium on Quality Electronic Design (ISQED'16)*, pp. 63–68, 2016.

[7] X. Zhao, S. Ma, Y. Liu, L. Eeckhout, and Z. Wang, "A low-cost conflict-free noc for gpgpus," in *Proceedings of the 53rd Annual Design Automation Conference*, p. 34, ACM, 2016.

[8] X. Zhao, S. Ma, C. Li, L. Eeckhout, and Z. Wang, "A heterogeneous low-cost and low-latency ring-chain network for gpgpus," in *34th IEEE International Conference on Computer Design (ICCD'16)*, pp. 472–479, 2016.

[9] C.-F. Tseng, C.-S. Liu, C.-H. Wu, and D. Yu, "Info (wafer level integrated fan-out) technology," in *66th IEEE Electronic Components and Technology Conference (ECTC'16)*, pp. 1–6, 2016.

[10] M. Matsuo, N. Hayasaka, K. Okumura, E. Hosomi, and C. Takubo, "Silicon interposer technology for high-density package," in *50th IEEE Electronic Components and Technology Conference (ECTC'00)*, pp. 1455–1459, 2000.

[11] R. Chaware, K. Nagarajan, and S. Ramalingam, "Assembly and reliability challenges in 3d integration of 28nm fpga die on a large high density 65nm passive interposer," in *62nd IEEE Electronic Components and Technology Conference*, pp. 279–283, 2012.

[12] JEDEC, "High bandwidth memory (hbm) dram," *JESD235*, 2013.

[13] S. Hou, W. C. Chen, C. Hu, C. Chiu, K. Ting, T. Lin, W. Wei, W. Chiou, V. J. Lin, V. C. Chang, *et al.*, "Wafer-level integration of an advanced logic-memory system through the second-generation cowos technology," *IEEE Transactions on Electron Devices*, vol. 64, no. 10, pp. 4071–4077, 2017.

[14] N. E. Jerger, A. Kannan, Z. Li, and G. H. Loh, "Noc architectures for silicon interposer systems: Why pay for more wires when you can get them (from your interposer) for free?," in *Proceedings of the 47th annual IEEE/ACM International Symposium on Microarchitecture (MICRO'14)*, pp. 458–470, 2014.

[15] B. Banijamali, C.-C. Chiu, C.-C. Hsieh, T.-S. Lin, C. Hu, S.-Y. Hou, S. Ramalingam, S.-P. Jeng, L. Madden, and D. C. Yu, "Reliability evaluation of a cowos-enabled 3d ic package," in *63rd IEEE Electronic Components and Technology Conference (ECTC'13)*, pp. 35–40, 2013.

[16] J. H. Lau, "The future of interposer for semiconductor ic packaging," *Chip Scale Rev*, vol. 18, no. 1, pp. 32–36, 2014.

[17] L. Li, P. Su, J. Xue, M. Brillhart, J. Lau, P. Tzeng, C. Lee, C. Zhan, M. Dai, H. Chien, *et al.*, "Addressing bandwidth challenges in next generation high performance network systems with 3d ic integration," in *62nd IEEE Electronic Components and Technology Conference (ECTC'12)*, pp. 1040–1046, 2012.

[18] K. Saban, "Xilinx stacked silicon interconnect technology delivers breakthrough fpga capacity, bandwidth, and power efficiency," *Xilinx, White Paper*, vol. 1, pp. 1–10, 2011.

[19] "High-bandwidth memory (hbm) reinventing memory technology." `www.amd.com/Documents/High-Bandwidth-Memory-HBM.pdf`, Aug 2016.

[20] NVIDIA, "Nvidia tesla p100: The most advanced data center accelerator." `https://www.nvidia.com/en-us/data-center/tesla-p100/`.

[21] D. Abts, N. D. Enright Jerger, J. Kim, D. Gibson, and M. H. Lipasti, "Achieving predictable performance through better memory controller placement in many-core cmps," *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3, pp. 451–461, 2009.

[22] J. De Vos, L. Bogaerts, T. Buisson, C. Gerets, G. Jamieson, K. Vandersmissen, A. La Manna, and E. Beyne, "Key elements for sub-50μm pitch micro bump processes," in *63rd IEEE Electronic Components and Technology Conference*, pp. 1122–1126, 2013.

[23] B. Williams, D. Florence, H. Dalal, K. Gunturu, M. Nelson, and C. Belisle, "Rdl manufacturing for flip chip packaging," in *IEEE Workshop on Microelectronics and Electron Devices (WMED'05)*, pp. 28–31, 2005.

[24] H. Li, H. Chua, F. Che, A. D. Trigg, K. Teo, and S. Gao, "Redistribution layer (rdl) process development and improvement for 3d interposer," in *13th IEEE Electronics Packaging Technology Conference (EPTC'11)*, pp. 341–344, 2011.

[25] "Samsung begins mass producing world's fastest dram based on newest high bandwidth memory (hbm) interface." `https://news.samsung.com/global/samsung-begins-mass-producing-worlds-fastest-dram-based-on-newest-high-bandwidth-memory-hbm-interface`, Jan 2016.

[26] A. Kannan, N. E. Jerger, and G. H. Loh, "Enabling interposer-based disintegration of multi-core processors," in *Proceedings of the 48th annual IEEE/ACM International Symposium on Microarchitecture (MICRO'15)*, pp. 546–558, 2015.

[27] E. T. Ogawa, K.-D. Lee, V. A. Blaschke, and P. S. Ho, "Electromigration reliability issues in dual-damascene cu interconnections," *IEEE Transactions on reliability*, vol. 51, no. 4, pp. 403–419, 2002.

[28] M. Khan and M. S. Kim, "Damascene process and chemical mechanical planarization," 2015.

[29] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, 2012.

[30] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–503, 2016.

[31] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[32] K. Arulkumaran, A. Cully, and J. Togelius, "Alphastar: An evolutionary computation perspective," *arXiv preprint arXiv:1902.01724*, 2019.

[33] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[34] Python, "The python tutorial." `https://docs.python.org/release/3.7.3/tutorial/index.html`.

[35] N. Jiang, J. Balfour, D. U. Becker, B. Towles, W. J. Dally, G. Michelogiannakis, and J. Kim, "A detailed and flexible cycle-accurate network-on-chip simulator," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS'13)*, pp. 86–96, 2013.

[36] Y. Kim, W. Yang, and O. Mutlu, "Ramulator: A fast and extensible dram simulator.," *Computer Architecture Letters*, vol. 15, no. 1, pp. 45–49, 2016.

[37] D. U. Lee, K. W. Kim, K. W. Kim, K. S. Lee, S. J. Byeon, J. H. Kim, J. H. Cho, J. Lee, and J. H. Chun, "A 1.2 v 8 gb 8-channel 128 gb/s high-bandwidth memory (hbm) stacked dram with effective i/o test circuits," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 1, pp. 191–203, 2015.

[38] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic, "Dsent-a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *6th IEEE/ACM International Symposium on Networks on Chip (NoCS'12)*, pp. 201–210, 2012.

[39] K. Hsieh, E. Ebrahimi, G. Kim, N. Chatterjee, M. O'Connor, N. Vijaykumar, O. Mutlu, and S. W. Keckler, "Transparent offloading and mapping (tom): Enabling programmer-transparent near-data processing in gpu systems," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 204–216, 2016.

[40] J. Yin, Z. Lin, O. Kayiran, M. Poremba, M. S. B. Altaf, N. Enright Jerger, and G. Loh, "Modular routing design for chiplet-based systems," in *International Symposium on Computer Architecture*, 2018.

[41] Xilinx, "Cost and transceiver optimized fpgas." `https://www.xilinx.com/products/silicon-devices/fpga/artix-7.html`.

[42] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," in *IEEE International Symposium on Workload Characterization (IISWC'09)*, pp. 44–54, 2009.

[43] NVIDIA, "Cuda code samples." `https://developer.nvidia.com/cuda-code-samples`, 2018.

[44] W. J. Dally, "Express cubes: improving the performance of k-ary n-cube interconnection networks," *IEEE Transactions on Computers*, vol. 40, no. 9, pp. 1016–1023, 1991.

[45] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," in *40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007)*, pp. 172–182, 2007.

[46] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, "Express cube topologies for on-chip interconnects," in *IEEE International Symposium on High Performance Computer Architecture (HPCA'09)*, pp. 163–174, 2009.

[47] K. H. Kim, R. Boyapati, J. Huang, Y. Jin, K. H. Yum, and E. J. Kim, "Packet coalescing exploiting data redundancy in gpgpu architectures," in *Proceedings of the International Conference on Supercomputing (ICS'17)*, p. 6, ACM, 2017.