Photonics-Aware Planning-Guided Automated Electrical Routing for Large-Scale Active Photonic Integrated Circuits

Hongjian Zhou[1], Haoyu Yang[2], Nicholas Gangi[3], Bowen Liu[3], Meng Zhang[3]
Haoxing Ren[2], Xu Wang[4], Rena Huang[3], Jiaqi Gu[1†]
[1]Arizona State University, [2]NVIDIA, [3]Rensselaer Polytechnic Institute, [4]Cadence

†*jiaqigu@asu.edu*

Abstract

The rising demand for AI training and inference, as well as scientific computing, combined with stringent latency and energy budgets, is driving the adoption of integrated photonics for computing, sensing, and communications. As active photonic integrated circuits (PICs) scale in device count and functional heterogeneity, physical implementation by manual scripting and ad-hoc edits is no longer tenable. This creates an immediate need for an electronic–photonic design automation (EPDA) stack in which physical design automation is a core capability. However, there is currently no end-to-end fully automated routing flow that coordinates photonic waveguides and on-chip metal interconnect. Critically, available digital VLSI and analog/custom routers are not directly applicable to PIC metal routing due to a lack of customization to handle constraints induced by photonic devices and waveguides. We present, to our knowledge, the first end-to-end routing framework for large-scale active PICs that jointly addresses waveguides and metal wires within a unified flow. We introduce a physically-aware global planner that generates congestion- and crossing-aware routing guides while explicitly accounting for the placement of photonic components and waveguides. We further propose a sequence-consistent track assignment and a soft guidance-assisted detailed routing to speed up the routing process with significantly optimized routability and via usage. Evaluated on various large PIC designs, our router delivers fast, high-quality active PIC routing solutions with fewer vias, lower congestion, and competitive runtime relative to manual and existing VLSI router baselines; on average it reduce via count by ∼99%, user-specified design rule violation by ∼98%, and runtime by 17×, establishing a practical foundation for EPDA at system scale.

1 Introduction

Photonic circuits offer intrinsic advantages, including low transmission loss and wide bandwidth that enable new classes of systems ranging from optical AI accelerators [1] to chiplet-scale interconnects and on-chip sensing arrays [2]. By co-integrating optical and electrical components, electronic–photonic integrated circuits (EPICs) combine high-speed photonic communication with energy-efficient electronic control. This hybrid system enables new capabilities across domains: low-latency AI hardware, terabit-scale inter-chip links, and near-sensor signal processing, all within a compact form factor. Fueled by advances in photonic device design and growing support from foundry process design kits (PDKs), EPICs are now being fabricated via multi-project wafers (MPWs) [3] and adopted by leading industry and academic efforts.

However, as EPICs scale to thousands of components, wires, and millimeter-scale routing paths, the physical design flow has become a dominant bottleneck. Current methodologies remain largely

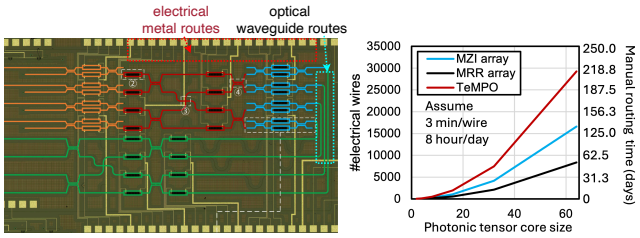

Figure 1: Electrical and optical routing in active PICs [4–7]. Manually routing a large number of electrical metal wires consumes days to weeks.

schematic-driven and manual: engineers hand-place devices, script waveguide paths to meet curvature and phase constraints, and manually route electrical metal to pads with careful attention to port alignment, via insertion, layer assignment, and thermal/optical interference. Commercial tools such as Synopsys OptoCompiler, Cadence Virtuoso, and GDSFactory provide schematic-driven layout environments and device libraries, but those semi-automated approaches still require extensive designer intervention for placement, routing, and layout closure. As a result, layout closure remains time-consuming, error-prone, and non-scalable for large EPICs.

Recent research has addressed parts of this problem, especially in waveguide-centric physical design. Tools such as PROTON [8], PLATON [9], and PlanarONoC [10] automate global waveguide routing using loss-aware pathfinding and grid-based methods. Others, such as Apollo [11] and LiDAR [12], have demonstrated scalable placement and curvy-aware waveguide routing with design-rule-compliant GDS generation. These tools represent real progress toward fabrication-ready electronic–photonic design automation (EPDA), particularly for passive photonic circuits.

Yet a critical gap remains: electrical routing automation for active PICs is not well supported, despite being essential to functionality and layout closure. In contrast to passive photonic circuits, active PICs, especially those computing circuits, integrate dense arrays of modulators, resonators, photodetectors, and phase shifters, requiring dense electrical interconnects for biasing, signaling, ground, or readout. As shown in Fig. 1, designers must carefully plan metal wires and I/O pad connections for active devices, find feasible routing solutions in limited metal layer resources, while ensuring compliance with both electronic design rules and photonic layout constraints. These steps account for a significant fraction of the total layout effort and are not scalable.

PIC electrical routing presents three fundamental challenges not addressed by conventional VLSI or analog routing tools: ❶ **Near-planar routing with high wire density and length**. Most photonic processes offer a few metal layers, e.g., 2 layers in AIM Photonics, and 8 layers in GlobalFoundries, with no layer-based routing

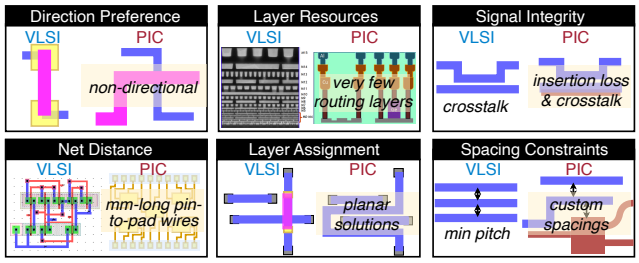

Figure 2: Compare electrical routing properties in VLSI circuits and active PICs.

directionality, which requires crossing-minimized wire planning and avoiding excessive via insertion. At the same time, large EPICs feature thousands of nets, many spanning millimeters across the die, leading to severe congestion and runtime challenges. ❷ **Electrical–optical interaction constraints.** Routing must avoid long, co-parallel overlap between metal and waveguides to limit thermal coupling and optical loss. Additionally, designers frequently impose custom isolation margins, beyond minimum spacing design rules, near critical photonic structures such as modulators and resonators. ❸ **Packaging-aware pad assignment and breakout.** PICs must respect fixed electrical and optical I/O pad locations, often driven by fiber array pitch or wirebonding constraints. This introduces geometric constraints on I/O breakout and enforces routing order preservation along chip edges.

Despite its critical role, electrical routing in EPICs remains a manual, fragmented process, handled independently from waveguide routing and absent from any integrated EPDA flow.

To address this gap, we propose the first end-to-end framework for large-scale active photonic integrated circuits that integrates electrical routing and waveguide routing into a unified, physically-aware flow. We introduce a global electrical routing planner that is congestion- and DRC-aware, explicitly accounts for the placement of photonic components and waveguides, and minimizes crossing and via usage. It further performs waveguide-aware track assignment and applies a guidance-driven detailed router that preserves route quality under tight spacing and co-propagation constraints with significantly faster runtime than prior un-guided detailed router. Our key contributions are as follows:

- **End-to-End Automated Routing Framework for Active PICs** – We develop the first automated flow that jointly routes electrical metal and photonic waveguides, targeting manufacturable, layout-rule-compliant PIC physical designs.
- **Physically-Aware Global Planning** – We introduce a photonics- and crossing-aware electrical global routing that minimizes wire crossings, metal–waveguide interaction, and via usage, while honoring user-specified design constraints.
- **Soft Guidance-Assisted Detailed Routing** – Our detailed router follows global guides and enforces spacing constraints with optical components, achieving high route quality and runtime efficiency in the near-planar regime.
- We extensively evaluate our router on large-scale active PIC benchmarks, and our framework achieves lower congestion, fewer vias and crossings, and competitive runtime relative to manual design and adapted VLSI routers, marking a significant step toward scalable, fabrication-ready EPDA. *We will open-source codes and our active PIC benchmark suites.*

2 Preliminaries

Routing in PICs comprises two domains: waveguide routing and electrical routing. Waveguide routing interconnects component optical ports, whereas electrical routing connects the electrical pins of active devices (e.g., the control wires that drive Mach-Zehnder modulators). The principal challenges in waveguide routing lie in enforcing curvilinear structures and automatically crossing insertion while minimizing insertion loss, which has been addressed in work [12]. In this section, we primarily focus on electrical routing for PICs by delineating its distinctions from conventional routing formulations and unique challenges.

2.1 Understanding the Challenges of PIC Electrical Routing

Automated routing in active PICs involves two physically distinct but tightly coupled domains: **waveguide routing**, which connects optical ports through curvature- and phase-constrained paths in the lowest silicon layer, and **electrical routing**, which establishes electrical connectivity for control, biasing, and readout of active components. While recent work has focused on automating waveguide routing, electrical routing in active PICs remains largely manual.

Beyond the basic design rules, minimum wire spacing, wire width, and metal area, several factors distinguish PIC electrical routing from VLSI and analog IC routing, as shown in Fig. 2

Limited Routing Layers and No Direction Preference. Most foundry PIC platforms expose only a small number of metal layers—typically two in AIM Photonics and up to eight in GlobalFoundries. In practice, designers often choose to restrict usage to fewer layers to reduce fabrication cost (e.g., fewer masks) and to minimize via-induced parasitics such as resistance, capacitance, and thermal hotspots. Unlike digital VLSI, where metal layers are organized with orthogonal directionality (e.g., horizontal M2, vertical M3), PIC electrical routing lacks this structure. Electrical nets in active PICs are often used not for internal logic connections but for I/O, with device pins fanned out toward peripheral pads for wirebonding. These factors result in a near-planar routing regime: *limited layers, long nets, dense component regions, and unconstrained directions*. Routing thousands of such nets within this regime demands careful congestion management, via minimization, and global planning across the entire die—challenges that traditional VLSI routers are ill-equipped to handle.

Electrical-Photonic Interaction Constraints. While metal wires can legally pass over photonic devices if no electrical conflict exists (i.e., short circuits), sustained co-propagation above waveguides and excessive overlap with photonic components are often discouraged. Metal structures near optical paths can introduce absorption, reflection, and thermal coupling, which degrade insertion loss and tuning efficiency. To mitigate this, designers prioritize avoiding the lower-layer metal wires from overlapping with: active photonic devices (e.g., heaters, phase shifters, modulators); passive components (e.g., directional couplers, Y-branches), and previously routed waveguides with long coupling length. These components act as (soft) blockages, reducing available routing area and *exacerbating congestion in already near-planar layout.*

Non-Standard, User-Specified Spacing Rules. Beyond foundry-defined design rules, minimum spacing (e.g., 1 μm), width, and metal

density, PIC designers usually impose custom, often aggressive, spacing constraints near sensitive photonic elements, sometimes exceeding 10 μm, to suppress electrical crosstalk, thermal hotspots, and layout-induced parasitics. These user-specified constraints are circuit- and platform-specific, and are not captured by a generic design rule checker (DRC) in traditional EDA tools.

Long Wires and Routing Scalability. PICs frequently require millimeter-scale metal wires for connecting heaters, modulators, and detectors across large dies. Routing such long nets over a fine grid (as used in VLSI) becomes runtime expensive due to grid explosion, while coarse grids suffer from poor resolution and frequent failure in congested regions. This creates a **scalability bottleneck**: fast routing engines must balance grid granularity with physical realism, particularly under complex component-aware constraints.

Together, these challenges make PIC electrical routing incompatible with standard VLSI/analog routers. In this work, we address these challenges by building a co-routing framework from the ground up, tailored to the unique constraints of active PICs.

2.2 Related Work

Although prior work on PIC electrical routing is lacking, related routing problems have been extensively explored in adjacent domains such as PCB/MCM routing [13–15] and packaging routing [16–18]. For example, in redistribution-layer (RDL) routing, the number of usable layers is typically very limited [19], so obtaining a (near-)planar graph solution is often a design objective. In peripheral I/O routing [16, 17], external pads must be routed to internal bumps, which introduces a pad-assignment problem [18]. In this setting, pin distributions in packaging are relatively regular, and there are typically no large blockages. Consequently, many works [16, 17, 20] adopt minimum-cost maximum-flow (MCMF) formulations; the underlying bipartite/grid graph is inherently planar in the absence of obstacles, so the solution yields crossing-free global routing paths for each net. The widely-studied PCB routing problem is to route all the connections between the BGA of chip packages while satisfying various physical constraints. Besides, the use of vias is often restricted, and thus the routing becomes planar. Previous studies categorized the PCB routing problem into two types: (1) the escape routing problem [21, 22] and (2) the area routing problem [23, 24]. The escape routing problem is to route from the pads of BGAs to their array boundaries. The works [25, 26] carry out layer assignment either following pin escape or concurrently with it, thereby limiting the number of vias employed during routing. The area routing problem is to connect the previously escaped routes of BGAs. At this stage, most works focus on length matching [24, 27]. Despite these similarities (planarity, peripheral I/O), these methods cannot be directly applied to PIC electrical routing: the irregular distribution of photonic components prevents the straightforward construction of a crossing-free graph, and the routing must additionally account for routed waveguides and user-defined spacing rules.

2.3 Problem Formulation

In this paper, we present (to the best of our knowledge) the first automated flow that performs both waveguide and electrical routing for PICs. Concretely, we decouple the two subproblems: waveguide routing is first carried out using LiDAR [12], followed by electrical routing. Our technical contributions primarily concern the electrical routing stage. We now formally define the PIC electrical routing problem as follows:

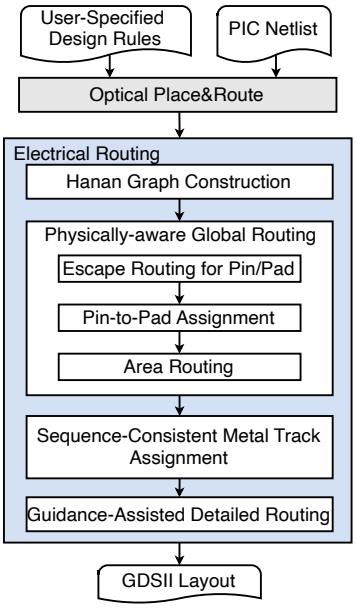

Figure 3: Algorithm flow of the proposed active PIC electrical routing framework.

PIC Electrical Routing. Given a set of electrical nets N_e, a set of placed photonic components, and the user-specified preferences, generate a routing solution for each $n_i \in N_e$, so that there are no design rule violations and the specified preferences are satisfied.

3 Automated Active PIC Electrical Routing

In this section, we present our metal routing algorithm for PIC. First, we give an overview of our proposed algorithm. Then, we detail the methods used in each stage of the algorithm.

3.1 Algorithm Overview

Figure 3 summarizes our routing framework. Given an input netlist consisting of optical and electrical nets, we first generate a manual placement and perform waveguide routing with LiDAR [12] to obtain the routed waveguide geometry. We then perform physically-aware global planning for the electrical nets, including pad assignment and the construction of routing guidance. The guidances are generated by two steps: (1) crossing-aware A* routing on Hanan grids that treats photonic components as blockages to obtain a (near-)planar solution, and (2) track assignment that respects the routed waveguides and user-specified design rules. Finally, these guidance cues are used to steer detailed routing. We next detail each component of the electrical routing stage.

3.2 Hanan Routing Grid Initialization

While direct A*-based detailed routing is commonly used in analog and custom design, it becomes **intractable and error-prone when applied directly to large PIC layouts**. Without global planning, A* often produces tangled paths, excessive crossings, or frequent ripup&reroute and routing failures in congested regions. To address these limitations, we adopt a physically-aware planning-guided

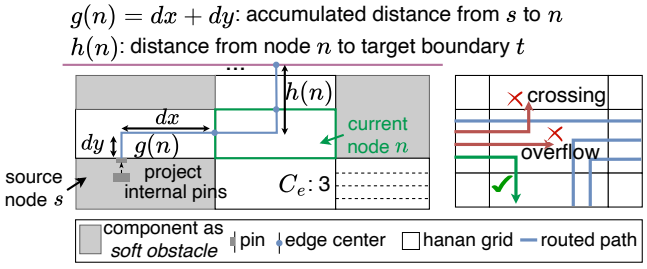

Figure 4: (a) Hanan routing grid construction and $g(n)$ and $h(n)$ cost calculation. (b) crossing check and capacity check.

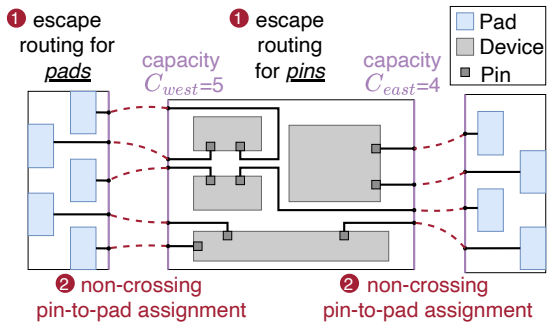

Figure 5: Illustration of ❶ pin/pad escape routing and ❷ pin-to-pad assignment. Escape net based on the distance to the boundary while satisfying the boundary edge capacity. Then assign pads based on the net order on the boundary to ensure a non-crossing assignment result.

routing flow , in which a global planning stage first computes a photonic-aware, capacity-constrained routing guide across multiple metal layers. This guide informs subsequent track assignment and detailed routing, improving routability and scalability.

To make metal routing photonic-aware and enable obstacle avoidance, we construct a customized Hanan grid over the layout, which encodes the layout geometry, metal-layer capacities, and photonic component locations, as described below.

The grid is initialized using the set of electrical pin coordinates and photonic component bounding boxes, which treats photonic components as *soft obstacles*. Pins located inside the component are first projected to the nearest grid boundary (along the shortest distance to the component boundary) before routing, as shown in Fig. 4. Unlike traditional Hanan grids, which embed nets as axis-aligned paths on graph edges, our grid supports **cell-based routing**, where paths *traverse between cell centers*, crossing boundaries orthogonally. For each Hanan grid edge e on layer ℓ, we compute a **capacity** $c_e = \lfloor \text{length}(e)/p_\ell \rfloor$, where p_ℓ is the minimum wire pitch of layer ℓ, specified by the process design kit (PDK). These capacities define the maximum number of parallel wire tracks that may legally traverse each edge, enabling congestion-aware multi-layer global metal planning. While the Hanan grid is defined in three dimensions, layers are decoupled; that is, grid graphs on distinct layers are independent, and no inter-layer edges are present.

3.3 Physically-Aware Global Routing

In the global routing stage, our goal is to generate crossing-aware routing guides. We propose a *crossing-aware A* planning engine* that enforces consistent net ordering across routing boundaries to minimize topological crossings. While conventional A* search is unaware of multi-net interactions and often produces intersecting routes, our method guarantees planarity by maintaining a global net ordering on each grid edge and pruning paths that would violate this ordering, inspired by prior work [28].

Crossing-Aware A* Planning Engine. The standard formulation of the node cost for the current node n in the A* search is given by: $f(n) = g(n) + h(n)$, where $g(n)$ denotes the accumulated cost from the source node s to the current node n, and $h(n)$ is the heuristic estimate of the cost from node n to the target node t. As illustrated in Fig. 4, when computing $g(n)$, we use the *actual pin location* at the source node, whereas for all other nodes we *measure distances with respect to the center point of the incident edge* since the actual tracks have not been assigned. The heuristic cost $h(n)$ is defined as the distance from the current node n to the *target bounding box boundary* t. Driven by the above-defined cost, our proposed crossing-aware A* search engine **tracks the net order that passes through each node edge in the routing grid** with a **net sequence list**: an ordered set of nets that traverse that edge. If two net path orders on the grid cell boundary become inconsistent, e.g., net1 and net2's orders are swapped on the incident and outgoing grid edges, those two electrical routing paths will cross each other and introduce a via. Figure 4 gives the example of crossing check and capacity check during routing. Therefore, during A* search for a given net, we try to find a planar (non-crossing) routing solution by maintaining a **consistent net order constraint at each grid node edge**, thereby generating a non-crossing topology.

While this net-sequence-based constraint guarantees the generation of a non-crossing guide for each individual net, the procedure is **inherently sequential and order-dependent**, and thus does not guarantee global optimality. Early-routed nets may consume critical routing corridors, block escape routes, and make later nets violate planarity, thus forcing via insertions and increasing metal layer usage. When no planar solution exists on the current layer, we permit **routing layer switching at endpoints only**, i.e., a net may route on a different layer from source to target, but *mid-path via insertions* are disallowed to preserve simplicity. If no feasible guides exist in any routing layers, the planning of this net will be skipped with no solution found, and the procedure for the rest of the nets will continue.

Multi-Stage Planning Flow: Pin/Pad Escape Routing → Pin-to-Pad Assignment → Area Routing. Since the metal wire crossings are mostly caused by *sub-optimal net-ordering and pin-to-pad assignment*, we propose to manage this complicated problem by decomposing global planning into three structured stages: (1) *escape routing*, (2) *pin-to-pad assignment*, and (3) *area routing*.

3.3.1 Pin/Pad Escape Routing. Rather than routing each electrical pin directly to its target pad, we first **route all pins to the boundary** of a common axis-aligned bounding box enclosing all active components, as shown in Fig. 5❶. This single, shared target (box boundary) avoids the net permutation chaos and thus significantly reduces path crossing risks, regularizes breakout topology, and simplifies pad assignment.

First, we determine the processing order for each pin. For a pin p, we compute its distance to the box boundary as the minimum of its distances to the four sides and process pins in non-descending order of this distance (nearer pins receive higher priority).

To avoid overloading any one side, we assign a side capacity to each edge of the bounding box, equal to the number of I/O pads available on that side. During escape routing, pins are assigned to bounding box edges under this capacity constraint, preventing escape congestion due to demand-resource imbalance and reducing downstream failure in the later area routing stage. And the A* heuristic cost $h(n)$ is set to the Manhattan distance from the current node to the boundary. We apply a similar *escape procedure to the pad arrays*, especially when they contain multiple rows or staggered placement, which ensures intra-pad-array planarity before completing routes from the box boundary to the assigned pads.

By performing escape routing on both the device pins and the I/O pads, we effectively build planar routing "bridges" from both ends. This strategy transforms the original complex many-to-many routing problem into a one-to-one mapping-and-route problem, and greatly simplifies the final connection stage.

3.3.2 Pin-to-Pad Assignment and Area Routing. A key advantage of performing escape routing first is that pin-to-pad assignment can be decided during routing. Otherwise, one must precompute a pin–to–pad mapping (e.g., via linear programming) prior to routing; without awareness of photonic components and keep-outs, such preassignment often fails to preserve planarity and complicates downstream routing.

After escaping all pad arrays and device pins to a common bounding-box boundary, we perform pin-to-pad assignment as illustrated in Fig. 5❷. The procedure is as follows: for each side of the boundary, we traverse the escaped pins following their orders on the boundary and assign them to pads on the same side in the corresponding order, respecting side capacities. Because the region outside the bounding box contains no active photonic components, the subsequent area routing from the boundary to the assigned pads can be carried out without crossings. For the area-routing stage, we reuse the crossing-aware A* search engine; the only difference from pin/pad escape routing is that the source–target paths are searched from the bounding-box boundary to the pad-array boundary.

3.4 Sequence-Consistent Track Assignment

After global routing, each net has a **sequence of Hanan grid nodes** that it traverses. The next critical step is to determine **specific track placement** within each node, while respecting the *net ordering* established on each edge during global routing. This stage produces a **fine-grained routing guidance** that ensures high routability, reduces crossings, and minimizes parallel metal–waveguide overlap. This procedure is outlined in Alg. 1.

Wire Segment Decomposition. We first decompose all routed paths into horizontal and vertical segments for each node they traversed. Track assignment proceeds in two passes: (1) Horizontal segments are assigned tracks, and the lengths of their vertical connections are updated; (2) Then, vertical segments are assigned, and the corresponding horizontal lengths are updated.

Panel-Based Joint Track Assignment. Rather than *locally* assigning tracks node by node, we introduce a **novel panel-based assignment** strategy with a *larger planning view*. A panel is a *window of contiguous grid nodes along a row or column*, containing all segments that may interact or compete for tracks. Within each panel, we leverage the net sequence information from global routing to perform *joint* track assignment. Specifically, we identify all segments that lie on the same row as the segment to be assigned, take the union of the grid nodes they occupy, and construct a panel for joint track assignment.

❶ *Constraint Graph Construction*: For all segments within the panel, we construct a constraint graph $G = (V, E)$, where each vertex represents a segment. As shown in Fig. 6, each vertex is

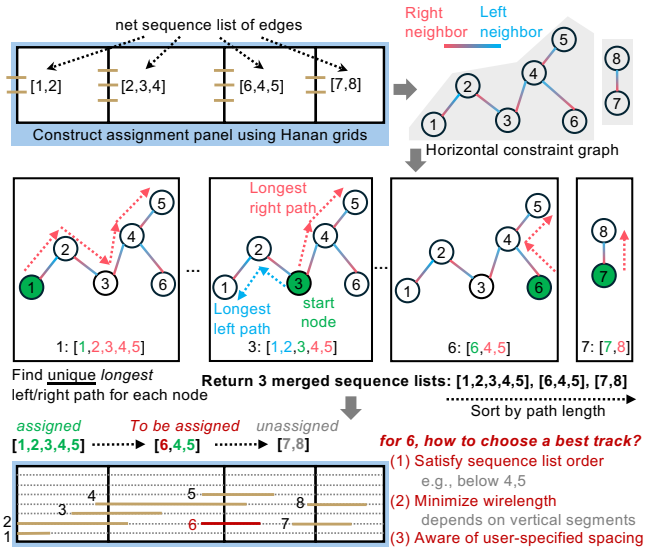

Figure 6: Overall track assignment procedure. An assignment panel and constraint graph are constructed using the Hanan grids and the net sequence list of each edge. Then, assign the segment using the merged sequence list according to wirelength/spacing considerations.

Algorithm 1 Track Assignment for Horizontal Segments

Input: Segments $\mathcal{S}$, net-sequence lists L, user-sepcified rules $\mathcal{R}$
Output: Per-net detailed routing guidance O

```
for each s ∈ S do
    if s.assigned then
        continue
    P ← BuildPanel(s, S)            ▷ using adjacent Hanan grids
    G = (V, E) ← BuildConflictGraph(P)
    M ← { MergedSequence(v, L, G) | v ∈ V }
    M ← SortSequencesByDepth(M)     ▷ deepest-first
    for each merged sequence m ∈ M do
        for each u ∈ OutsideToInsideOrder(m, P) do
            if u.assigned then
                continue
            AssignTrack(u, P, R)
            UpdateVerticalSegment(u)
            u.assigned ← true
O ← StitchGuidelines(S)
```

connected to left neighbor nodes via blue edges and connected to right neighbor nodes via red edges. In the horizontal constraint graph example in the figure, given the net sequence [2,3,4] and [6,4,5], node 4 has two left nodes 3 and 6, and one right node 5.

❷ *Merging Net-Sequences*: To identify the maximal mergeable net-sequence blocks, we find the longest left path (defined as the longest path that only contains edges that point to left neighbors) and longest right path for each node simply using a recursive algorithm. Then it forms the longest mergeable net-sequence by concatenating the longest left path, the start node itself, and the longest right path, e.g., for start node=3, we have [1,2] | [3] | [4,5]→[1,2,3,4,5]. After deduplication, we find all **unique longest mergeable net-sequences**. The merged sequences will be ranked by depth, with deeper ones assigned to tracks first. This **deepest-first ordering** ensures that subsequent inner segments have sufficient available tracks, reducing conflicts and resource contention.

❸ *Wirelength-minimized Spacing-honored Track Assignment*: In each *panel*, track positions are discretized using the *layer-specific metal wire pitch*. For each segment in a merged net-sequence list, we assign tracks according to the following **rules**: **(1) Order Constraint**: We select available tracks that satisfy the sequence list order, e.g., in the example given in the figure, only the 3 unoccupied tracks below the assigned segment 4 can be selected as candidate tracks for segment 6. **(2) Wirelength Minimization**: Different track assignment can potentially induce different wirelength, especially when the two connecting segments of the interested segment are on the same side. If both neighboring segments of the same net are pointing in the same direction, we will bias the track toward that direction to minimize wirelength and select the nearest available track. If neighbors lie on opposite sides, any track yields equivalent Manhattan length. **(3) Ouside-In Assignment**: Tracks are assigned from the outermost segments of the panel inward. This ordering simplifies resource allocation and prevents later conflicts. **(4) Pin Access and Spacing Constraints**: Segments that are directly connected to pins must lie within the pin's access window. Meanwhile, user-preferred wire spacing is enforced between segments; if insufficient space remains, spacing is reduced to the maximum feasible value. **(5) Optical Waveguide Interaction Check**: After assignment, we check each segment's parallel overlap with pre-routed waveguides. If the overlap exceeds a threshold L_{th}, the segment is shifted by at most one track to reduce unexpected metal-induced waveguide insertion loss; otherwise, the assignment is accepted.

Finally, the assigned segments are *stitched into continuous routing guidelines for each net*, which serve as input for the detailed routing stage, ensuring significantly faster routing and higher planar solution quality that respects both electrical and optical constraints.

3.5 Soft Guidance-Assisted Detailed Routing

After track assignment, fine-grained detailed routing is required to generate DRC-compliant metal paths that satisfy all pin access, spacing, and photonic constraints. While standard A*-based detailed routers can perform grid-based routing with electrical DRC checks, they **struggle in the PIC context** as they (1) cannot handle electrical-photonic interactions, as metal wires that overlap or run close to photonic components can degrade optical performance via absorption or thermal coupling; (2) cannot effectively follow global routing guides: track guidelines may become blocked due to pin placement constraints or already routed wires. **Hard on-guideline** enforcement often causes the search to lose the intended path and drift away, resulting in unnecessary detours or routing failure, as illustrated in Fig. 7.

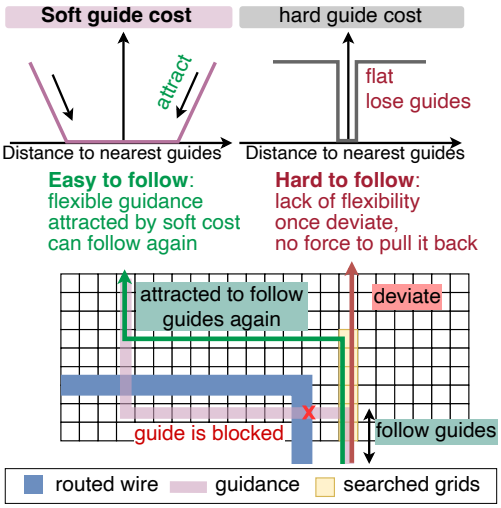

Figure 7: Proposed soft guidance provides a flexible mechanism to attract the searching node to follow the guidance even with blocked guidance and temporary search path deviation, and thus can accelerate the routing process.

Table 1: Active PIC benchmark information.

Benchmark	#Devices	#Optical Nets	#Electrical Nets	Die Size
Clements_8×8 [30]	132	87	71	$3600 \times 1800\ \mu m^2$
Clements_16×16 [30]	456	303	271	$7600 \times 3300\ \mu m^2$
Clements_32×32 [30]	1682	1119	1056	$14300 \times 6100\ \mu m^2$
ADEPT_8×8 [31]	232	127	127	$4500 \times 2000\ \mu m^2$
ADEPT_16×16 [31]	547	319	319	$7400 \times 3600\ \mu m^2$
ADEPT_32×32 [31]	1299	767	767	$100000 \times 6000\ \mu m^2$
GWOR_8×8 [32]	118	32	85	$3000 \times 3000\ \mu m^2$
GWOR_10×10 [32]	190	50	160	$5100 \times 5100\ \mu m^2$

To address these issues, we extend an open-source analog detailed router, `Anaroute` [29], with two key enhancements: **(1) Photonic Awareness**: We impose additional penalties on routing paths that overlap or are too close to photonic components and waveguides. This ensures metal paths respect photonic layout constraints. **(2) Soft Guidance for Track Following**: Instead of a hard "on-guideline" constraint, we propose a critical technique: **distance-based soft penalty** that increases as a wire deviates from the assigned guideline. This encourages the routing search to stay close to the global routing guidance, while still being flexible, allowing detours in congested/blocked regions. When a guideline segment is blocked by pin access or previously routed wires, the soft penalty allows the search to **temporarily deviate but then attracts the path back** toward the guideline once the blockage is cleared. Figure 7 illustrates this behavior: A hard-guideline search would cause the path to drift away and lose guidance, whereas soft guidance maintains path fidelity while providing flexibility, producing smoother, DRC-compliant routing that honors both electrical and photonic constraints.

4 Evaluation Results

The proposed framework for PIC electrical routing and waveguide routing is implemented in Python and C++. The global routing is

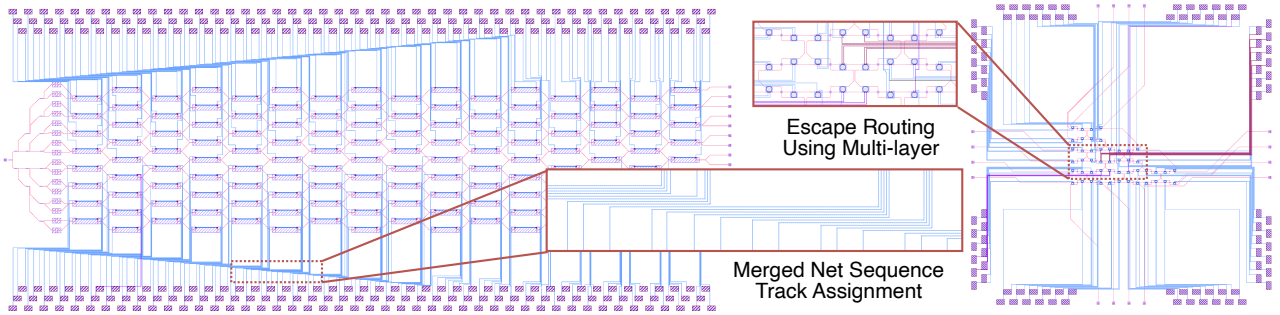

Figure 8: Visualization of our generated complete PIC layout (Clements_16×16 [5], GWOR_8×8 [32]) with automatically routed optical waveguides and electrical connections. The zoomed-in regions highlight the planar routing solution from our multi-layer escape routing and net-sequence-based track assignment algorithms.

Table 2: Comparisons of the used via number (#Via), used routing layer (#Layer), number of user-specified design rule violations (#USV), runtime (s), and the total wirelength (WL (*mm*)). ↓: lower is better.

Benchmark	Anaroute [33]					Anaroute* [33]					Our Proposed Router				
	#Via↓	#Layer↓	#USV↓	Runtime (s)↓	WL (*mm*)↓	#Via↓	#Layer↓	#USV↓	Runtime (s)↓	WL (*mm*)↓	#Via↓	#Layer↓	#USV↓	Runtime (s)↓	WL (*mm*)↓
Clements_8×8	1	2	105	45	42.99	2	3	23	114	45.86	0	1	2	33	46.94
Clements_16×16	69	3	879	2383	340.97	117	3	370	2324	348.69	0	1	4	207	360.51
Clements_32×32	159	3	2374	13235	2084.42	205	3	753	20093	2095.78	0	2	12	930	2124.23
ADEPT_8×8	8	3	178	212	95.58	16	3	74	481	97.2	0	1	2	65	100.7
ADEPT_16×16	54	3	674	1558	422.9	83	3	249	3184	425.98	0	1	22	245	440.56
ADEPT_32×32	402	3	1603	10696	2319.2	519	3	853	17523	2325.31	0	2	37	871	2347.02
GWOR_8×8	43	3	357	2494	181.8	64	3	248	3343	185.02	6	3	19	99	184.84
GWOR_10×10	173	3	2904	17281	561.20	228	3	1347	24960	566.72	2	3	26	316	567.5
Geo-mean	114	-	1134	5988	756.13	154	-	490	9003	761.32	1	-	16	346	771.53
ratio	1.000	-	1.000	1.000	1.000	1.351	-	0.432	1.504	1.007	0.008	-	0.014	0.058	1.020

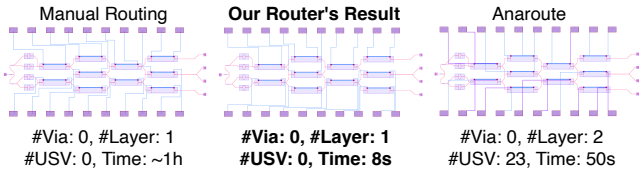

Figure 9: Comparison of manual and automated routing of (Clements_4×4 [5]). Our router achieves comparable results to manual routing while requiring substantially less time, while naive Anaroute fails to find a planar routing solution, and the routed metal wires cross the photonic components.

developed using Python, while the detailed routing is developed based on the router `Anaroute` in MAGICAL [33]. The waveguide routing is done by an open-source PIC waveguide router `LiDAR` [12]. All evaluations are conducted on a Linux server with a 128-core AMD EPYC 7763 CPU and 512 GB memory.

Benchmarks. We evaluate our router on large-scale active PIC benchmarks in computing and interconnect applications, derived from publicly available PIC designs [12]. For each design, we add electrical nets to the active components and connect their pins to the chip's peripheral I/O pads to enable active component control. We summarize the benchmark statistics in Table 1. For the photonic computing tensor cores, Clements [5] and ADEPT [31] circuits, we place the I/O pads along the top and bottom edges of the chip, while optical grating couplers are placed along the left and right edges. Compared to Clements, ADEPT contains more photonic components and nets. For the optical interconnect switches, GWOR [32], both I/O pads and grating couplers are distributed along all four sides of the chip, with sufficient clearance reserved for packaging and coupling alignment. All benchmark circuits contain three available metal layers and one silicon waveguide routing layer.

Baselines. Since no automated electrical router for PICs is currently available, we compare against an existing analog router `Anaroute` [33], which supports multiple metal layers and arbitrary routing directions within each layer, thereby covering the basic requirements of PIC electrical routing. Based on this, we further construct a *photonic-aware* variant as a second baseline (`Anaroute`*): during path searching, we assign penalties to grid cells that overlap with photonic components, discouraging routing on top of optical components. For both baselines, we use the *same netlist* produced after our escape routing and pad assignment. This isolates the routing algorithms from confounding factors in pad assignment; otherwise, a naive pad assignment can easily introduce unavoidable crossings and unfairly penalize the baselines.

4.1 PIC Electrical Routing Evaluation Results

Evaluation Metrics. Since our contributions focus on electrical routing for active PICs, we evaluate the quality of the metal routing solutions using the following metrics: **(1) Metal Layer Usage**: total number of metal layers used in routing, reflecting the planar routing capability of our routers, critical for PICs with limited metal layer stacks. **(2) Via Usage**: total number of vias, which impact signal integrity, parasitics, and reliability. **(3) User-Specified Rule Violations (USVs)**: designer-defined constraints beyond PDK rules, including: (a) Enlarged wire spacing requirements (e.g.,$0.5\mu m \rightarrow 3\mu m$); (b) Metal traces should try to avoid crossing over photonic components; (c) Metal wires at M1 should avoid long parallel overlap with waveguides ($> 20\mu m$ overlapping length counts as a violation); (d) Metal wires at different layers should also avoid significant overlapping due to crosstalk concerns. *All DRC and USV checks are implemented using KLayout's DRC tool.* **(4) Runtime and Wirelength**: We report the total routing runtime and electrical wirelength. For the low-speed PIC benchmarks

Table 3: Our router can support high-quality electrical routing for `Clements_8×8` across different metal pitch settings.

Metrics	pitch=5μm	pitch=10μm	pitch=15μm	pitch=20μm
#Via	0	0	0	2
#Layer	1	1	2	3
#USV	2	1	4	6
Runtime (s)	33	36	35	38
WL (mm)	46.94	47.87	48.56	50.01

considered (<1 GHz), electrical nets connecting I/O pads to pins primarily carry DC bias to program active photonic devices rather than high-speed signals as in VLSI sequential logic or RF ICs, so *timing is not a critical metric for active low-speed PIC metal routing*. **Wirelength serves only as an indirect hint of routability, rather than a measure of circuit performance**. Differences in wirelength shown later do not imply differences in PIC functionality or performance; therefore, it is not a focused metric.

Main Results. Table 2 reports quantitative comparisons of the routing results, and Fig. 8 visualizes the layouts produced by our router. With crossing-aware global routing, our approach consistently requires fewer metal layers and significantly fewer vias, while better satisfying designer constraints. Compared to the naïve baseline, we achieve a ∼**99%** reduction in via usage, a ∼**98%** decrease in #USVs, and a **17**× runtime reduction, yielding high-quality electrical routing under tight turnaround. In particular, on `Clements` and `ADEPT`, despite the large number of active devices, (1) We obtain **planar** solutions that complete electrical routing using a single metal layer (M1). For `GWOR`, pins must escape to the periphery, which increases the likelihood of crossings during escape routing. Our router uses three metal layers, yet still requires far fewer vias than baselines. This indicates that only a small fraction of nets climb to upper layers, and the majority remain on M1, yielding a *near-planar* solution; the few nets that use vias can be manually adjusted if a planar layout is enforced. (2) A substantial portion of the **speedup** comes from our routing *guidance*, which restricts detailed routing to narrow corridors around pre-prepared guidelines and thus greatly shrinks the search space. With guidance, we employ a fine grid step of 1 μm; the naive baseline must relax to 2 μm to keep runtime manageable, as nets may exceed 1mm in length, otherwise its runtime grows substantially. We also compare to manual routing in Fig. 9. Our router achieves comparable layout quality with orders-of-magnitude speedup (1h→8s). Figure 10 shows the runtime breakdown of our router, where detailed routing dominates the total runtime. (3) Thanks to our *track assignment*, we can **meet designer requirements** in global routing by enlarging inter-segment spacing and proactively avoiding inter-layer overlaps, thereby reducing post detailed routing USVs. In contrast, the baseline lacks spacing control and routes each net independently; it honors the basic design rules but cannot satisfy higher-level designer preferences. Note that our results incur only a small increase in wirelength, which is not the primary objective in this context.

4.2 Robust Support of Different Wire Pitch

Since our framework provides fine-grained control over wire spacing, we can evaluate robustness under different *wire pitch* requirements. We sweep the spacing constraints and update the corresponding `KLayout` DRC rules (see Table 3). Across all settings, our router successfully honors the specified spacing requirements while

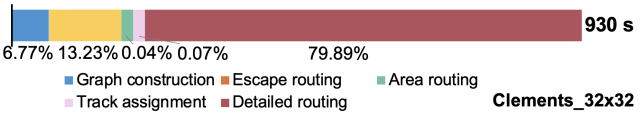

Figure 10: Runtime breakdown of our router.

Table 4: *Soft* guidance outperforms hard guidance with significantly higher routing quality and faster runtime.

Metrics	Clement_8×8		ADEPT_8×8		GWOR_8×8	
	Hard	***Soft***	Hard	***Soft***	Hard	***Soft***
#Via	0	0	47	0	102	6
#Layer	1	1	2	1	3	3
#USV	2	2	36	2	179	19
Runtime (s)	37	33	923	65	1852	99
WL (mm)	46.84	46.94	97.75	100.70	183.20	184.84

maintaining low #USVs and stable runtime. A larger wire pitch reduces the number of available tracks in each routing channel formed by photonic components, which forces more nets onto additional metal layers. As a result, the overall metal layer usage increases. Note that the reported #Vias remains 0 in some cases because layer transitions inserted directly at pins/pads to move a net onto another metal layer are not counted; we only count vias introduced along routed paths during the detailed routing stage.

4.3 Effectiveness of Soft Guidance Cost

A key feature of our framework is the *soft guidance cost*, which endows detailed routing with error-correction capability. To validate this property, we compare against a *hard guidance* baseline, i.e., we only raise the priority of routing grids that touch the guidance. From Table 4, the guidance mechanism has a substantial impact on final routing quality. The *hard guidance* cost fails to fully leverage the global-routing guidelines and often degenerates into a naive detailed router. There are two main reasons: (1) **Grid-step sensitivity.** Our guidelines are fine-grained. If the search frontier does not touch guideline cells (because the guidance may not be the shortest path, to avoid components or vias), the hard-boost rule may never be triggered, causing the search to miss guidance entirely. (2) **Imperfect/Conflicting guidelines.** Guidelines are not perfect due to pin-location constraints and local interactions; when a guideline collides with already routed nets, the hard-boost quickly becomes ineffective and can even hurt quality. On ADEPT and GWOR, we observe more crossings and longer runtimes with hard guidance. This is because, once the search deviates from the guidance and reconnects through a cross-layer neighbor, additional vias are introduced. The deviation also disrupts search continuity, leading to unnecessary expansions. In contrast, soft guidance provides "error-correction": it steers the A* search back onto, or near, the guidance corridor and then proceeds toward the target node.

5 Conclusion

We introduce the first photonics-aware electrical router for large-scale active photonic integrated circuits, completing the missing piece of the PIC physical design automation flow and enabling the first truly end-to-end automated layout solution. Our framework integrates novel photonics-aware global routing, sequence-consistent track assignment, and soft guidance–assisted detailed routing to produce a high-quality PIC layout with minimal layer/via usage while flexibly honoring designer preferences. Crucially, we show

that PIC electrical routing is not a simple extension of VLSI routing but requires new strategies that respect unique electronic–photonic interactions. Our customized router achieves superior layout quality, scalability, and up to 17× speedup over existing EDA tools on large-scale PIC benchmarks. This work lays the foundation for scalable, fabrication-ready EPDA, accelerating EPIC design cycles and paving the way for the next generation of manufacturable, energy-efficient, and high-performance electronic–photonic systems.

References

[1] Shanshan Yu Shiyue Hua, Erwan Divita et al. An integrated large-scale photonic accelerator with ultralow latency. *Nature*, 2025.

[2] Yuyang Wang, Songli Wang, Robert Parsons, Asher Novick, Vignesh Gopal, Kaylx Jang, Anthony Rizzo, Chia-Pin Chiu, Kaveh Hosseini, Tim Tri Hoang, Sergey Shumarayev, and Keren Bergman. Silicon photonics chip i/o for ultra high-bandwidth and energy-efficient die-to-die connectivity. In *Proc. CICC*, pages 1–8, 2024.

[3] Roger Helkey. Introducing aim photonics: Advancing photonic integrated circuit (pic) manufacturing, 2017.

[4] Chenghao Feng, Jiaqi Gu, Hanqing Zhu, Zhoufeng Ying, Zheng Zhao, et al. A compact butterfly-style silicon photonic–electronic neural chip for hardware-efficient deep learning. *ACS Photonics*, 9(12):3906–3916, 2022.

[5] Yichen Shen, Nicholas C. Harris, Scott Skirlo, et al. Deep learning with coherent nanophotonic circuits. *Nature Photonics*, 2017.

[6] Alexander N. Tait, Thomas Ferreira de Lima, Ellen Zhou, et al. Neuromorphic photonic networks using silicon photonic weight banks. *Sci. Rep.*, 2017.

[7] Meng Zhang, Dennis Yin, Nicholas Gangi, Amir Begović, Alexander Chen, Zhaoran Rena Huang, and Jiaqi Gu. Tempo: efficient time-multiplexed dynamic photonic tensor core for edge ai with compact slow-light electro-optic modulator. *Journal of Applied Physics*, 135(22), 2024.

[8] Anja Boos, Luca Ramini, Ulf Schlichtmann, and Davide Bertozzi. Proton: An automatic place-and-route tool for optical networks-on-chip. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 138–145. IEEE, 2013.

[9] Anja von Beuningen and Ulf Schlichtmann. Platon: A force-directed placement algorithm for 3d optical networks-on-chip. In *Proceedings of the 2016 on International Symposium on Physical Design*, pages 27–34, 2016.

[10] Yu-Kai Chuang, Kuan-Jung Chen, Kun-Lin Lin, Shao-Yun Fang, Bing Li, and Ulf Schlichtmann. Planaronoc: concurrent placement and routing considering crossing minimization for optical networks-on-chip. In *Proceedings of the 55th Annual Design Automation Conference*, pages 1–6, 2018.

[11] Hongjian Zhou, Haoyu Yang, Gangi Nicholas, Haoxing Ren, Huang Rena, and Jiaqi Gu. Apollo: Automated Routing-Informed Placement for Large-Scale Photonic Integrated Circuits. In *Proc. ICCAD*, 2025.

[12] Hongjian Zhou, Keren Zhu, and Jiaqi Gu. Lidar: Automated curvy waveguide detailed routing for large-scale photonic integrated circuits. In *Proc. ISPD*, 2025.

[13] Shih-Ting Lin, Hung-Hsiao Wang, Chia-Yu Kuo, Yolo Chen, and Yih-Lang Li. A complete pcb routing methodology with concurrent hierarchical routing. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 1141–1146. IEEE, 2021.

[14] Ting-Chou Lin, Devon Merrill, Yen-Yi Wu, Chester Holtz, and Chung-Kuan Cheng. A unified printed circuit board routing algorithm with complicated constraints and differential pairs. In *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, pages 170–175, 2021.

[15] Qinghai Liu, Qinfei Tang, Jiarui Chen, Chuandong Chen, Ziran Zhu, Huan He, Jianli Chen, and Yao-Wen Chang. Disjoint-path and golden-pin based irregular pcb routing with complex constraints. In *Proc. DAC*, 2023. doi: 10.1109/DAC56929.2023.10247728.

[16] Jia-Wei Fang, I-Jye Lin, Ping-Hung Yuh, Yao-Wen Chang, and Jyh-Herng Wang. A routing algorithm for flip-chip design. In *ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005.*, pages 753–758. IEEE, 2005.

[17] Jia-Wei Fang, I-Jye Lin, Yao-Wen Chang, and Jyh-Herng Wang. A network-flow-based rdl routing algorithmz for flip-chip design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(8):1417–1429, 2007.

[18] Tao-Chun Yu, An-Jie Shih, and Shao-Yun Fang. Flip-chip routing with i/o planning considering practical pad assignment constraints. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 27(8):1921–1932, 2019.

[19] Jia-Wei Fang and Yao-Wen Chang. Area-i/o flip-chip routing for chip-package co-design. In *2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 518–522. IEEE, 2008.

[20] Po-Wei Lee, Hsu-Chieh Lee, Yuan-Kai Ho, Yao-Wen Chang, Chen-Feng Chang, I-Jye Lin, and Chin-Fang Shen. Obstacle-avoiding free-assignment routing for flip-chip designs. In *Proceedings of the 49th Annual Design Automation Conference*, pages 1088–1093, 2012.

[21] Seong-I Lei and Wai-Kei Mak. Optimizing pin assignment and escape routing for blind-via-based pcbs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(2):246–259, 2015.

[22] Hui Kong, Tan Yan, and Martin DF Wong. Optimal simultaneous pin assignment and escape routing for dense pcbs. In *2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 275–280. IEEE, 2010.

[23] Tan Yan and Martin DF Wong. Recent research development in pcb layout. In *2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 398–403. IEEE, 2010.

[24] Jin-Tai Yan and Zhi-Wei Chen. Obstacle-aware length-matching bus routing. In *Proceedings of the 2011 international symposium on Physical design*, pages 61–68, 2011.

[25] Muhammet Mustafa Ozdal and Martin DF Wong. Algorithms for simultaneous escape routing and layer assignment of dense pcbs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(8):1510–1522, 2006.

[26] Jin-Tai Yan. Layer assignment of buses and nets with via-count constraint in high-speed pcb designs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 38(3):512–525, 2018.

[27] Tan Yan and Martin DF Wong. Bsg-route: A length-constrained routing scheme for general planar topology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(11):1679–1690, 2009.

[28] Min-Hsuan Chung, Je-Wei Chuang, and Yao-Wen Chang. Any-angle routing for redistribution layers in 2.5 d ic packages. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2023.

[29] Hao Chen, Mingjie Liu, Biying Xu, Keren Zhu, Xiyuan Tang, Shaolan Li, Yibo Lin, Nan Sun, and David Z. Pan. Magical: An open- source fully automated analog ic layout system from netlist to gdsii. *IEEE Design & Test*, 38(2):19–26, 2021.

[30] William R. Clements, Peter C. Humphreys, Benjamin J. Metcalf, et al. Optimal Design for Universal Multiport Interferometers. *Optica*, 2018.

[31] Jiaqi Gu, Hanqing Zhu, Chenghao Feng, Zixuan Jiang, Mingjie Liu, Shuhan Zhang, Ray T. Chen, and David Z. Pan. Adept: Automatic differentiable design of photonic tensor cores. In *Proc. DAC*, 2022.

[32] Xianfang Tan, Mei Yang, Lei Zhang, Yingtao Jiang, and Jianyi Yang. A generic optical router design for photonic network-on-chips. *Journal of Lightwave Technology*, 30(3):368–376, 2011.

[33] Biying Xu, Keren Zhu, Mingjie Liu, Yibo Lin, Shaolan Li, Xiyuan Tang, Nan Sun, and David Z Pan. Magical: Toward fully automated analog ic layout leveraging human and machine intelligence. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8. IEEE, 2019.