DSR: A Systematic Approach for Efficient Double-sided Signal Routing

Jianqing Chen[†1], Zhenkun Lin[†2], Xun Jiang[2], Genggeng Liu[*1], Yibo Lin[2], Gang Du[2]
[1]College of Computer and Data Science, Fuzhou University, Fuzhou, China
[2]School of Integrated Circuits, Peking University, Beijing, China

***Abstract*—The emergence of back-side interconnects aims to sustain the continued scaling of semiconductor technology. To extend existing back-end tools, netlist planning has been introduced to transform single-sided netlists into double-sided ones, thereby exploring the potential of utilizing bridging cells for double-sided signal routing. However, the lack of a native double-sided routing approach that fully leverages both front-side and back-side resources hinders the effective handling of complex systematic requirements. In light of this, we propose a native double-sided signal routing approach DSR for the first time, which realizes efficient cross-layer path selection in 3D routing space by unified modeling of front-side and back-side resources. We develop a native double-sided global routing algorithm that jointly considers resource allocation and bridging cell insertion, guided by delay models for performance optimization. Under the guidance of global routing, we further extend the double-sided routing graph and incorporate delay-aware mechanisms to enhance resource allocation and routing quality in detailed routing. Experimental results demonstrate that, compared with existing works, the proposed approach achieves significant improvements in delay and runtime, while maintaining wirelength and eliminating Design Rule Violations (DRVs).**

***Index Terms*—Back-Side Interconnects, Double-Sided Signal Routing, Delay Optimization**

I. Introduction

Back-side Power Delivery Network (BSPDN) [1] mitigates IR-drop by transferring power delivery to the Back-Side (BS) of the chip, thereby releasing Front-Side (FS) routing resources, improving layout efficiency, and reducing power loss. With FS layers becoming increasingly congested and timing constraints tightening, both academia [2]–[10] and industry [11], [12] have begun to explore the utilization of BS resources for routing, including signal, power, and clock net. As shown at the bottom of Fig. 1, double-sided signal routing employs bridging cells in the device layer to connect FS and BS metal layers, taking advantage of the lower resistance and parasitic capacitance of BS metals. This enables faster transmission and improved performance for timing-critical nets. However, the lack of native double-sided signal routing approach severely restricts the full development and utilization of double-sided routing resources.

Current studies on double-sided routing mainly extend conventional single-sided approaches. As shown in Fig. 1, prior works [4], [5] adopt netlist planning to transform single-sided netlists into double-sided ones. Their basic assumption is that nets with larger bounding boxes are more likely to be timing-critical, and thus preferentially assigned to BS routing resources. Afterward, single-sided routers are independently applied to the front and back sides. However, identifying timing-critical nets before routing is inherently difficult, which prevents full utilization of low-resistance BS layers and higher FS layers. This independent routing strategy often leads to Design Rule Violations (DRVs). To improve legality and efficiency, previous work [13] proposed a post-routing Engineering Change Order (ECO) strategy. Nevertheless, ECO flows significantly degrade design efficiency and still fail to address timing optimization requirements. Moreover, existing methods lack flexibility in handling stringent timing constraints and other systematic requirements, which severely limits the collaborative utilization efficiency of double-sided routing resources.

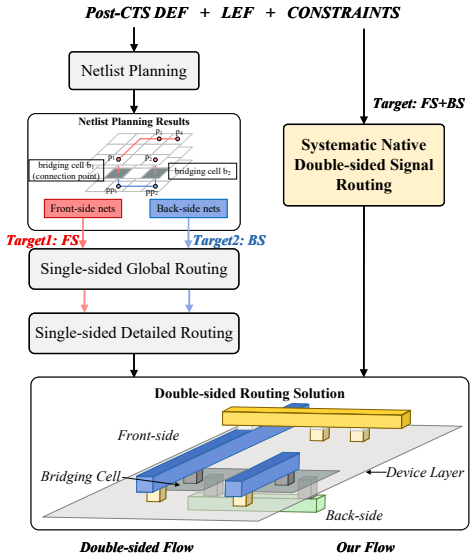


Fig. 1 Signal routing with double-sided metal layers.

Given these complex requirements, double-sided signal routing research still lags behind advanced technology scaling. Signal routing needs to systematically optimize multiple critical objectives, including wirelength, timing performance, design rules, and resource utilization. The indiscriminate insertion of bridging cells, when precise identification of timing-critical nets is challenging, may introduces additional legalization issues, wirelength, parasitic resistance, and DRVs, which prevents efficient utilization of double-sided resources and ultimately impairs overall chip design efficiency. This motivates us to

† These authors contributed equally to this work.
* Corresponding author. Email:liugenggeng@fzu.edu.cn

This work was supported in part by the National Natural Science Foundation of China under Grant 62372109 and in part by the Fujian Natural Science Funds under Grant 2023J06017.

explore a solution that unifies double-sided routing spaces while systematically considering the routing objectives. To the best of our knowledge, no prior work has proposed an efficient approach to unify the double-sided space to solve the systematic requirements of double-sided signal routing.

To address the aforementioned challenges, we propose for the first time an efficient double-sided signal routing approach DSR, as shown in the flow on the right of Fig. 1. DSR integrates FS and BS resources into a unified 3D routing space, overcoming the limitations of static netlist planning. Within this unified space, DSR systematically considers wirelength, DRVs, path delay, and bridging cell legalization, dynamically determining routing paths and bridging cell insertions. The proposed approach aims to efficiently coordinate double-sided resources, fully leveraging the advantages of low-resistance metal layers. The main contributions of this work are summarized as follows:

- To the best of our knowledge, we present the first systematic approach for efficient double-sided signal routing, constructing a unified 3D space that enables dynamic routing and resource allocation across double-sided.
- We develop a native double-sided global routing algorithm guided by a 3D routing delay model, which jointly considers resources on both sides and bridging cell insertion.
- We extend the double-sided routing graph and incorporate delay awareness to develop an efficient double-sided detailed router, guided by the global double-sided routing, to achieve systematic optimization.
- Experimental results based on the ASAP7 PDK show that the proposed approach effectively improves critical-path delay and resource utilization. While maintaining wirelength and eliminating DRVs, it achieves an average total delay reduction of 17.4%, a maximum delay reduction of 18.1%, and an average runtime improvement of 7.2%.

II. Preliminaries

A. Signal Routing

In Very-Large-Scale-Integrated (VLSI) circuit design, signal routing is a key step after Clock Tree Synthesis (CTS), producing legal and efficient interconnections for all nets across multiple metal layers. Due to its complexity, routing is divided into two stages: Global Routing (GR) and Detailed Routing (DR). Guided by Steiner trees, GR generates coarse routing guides for connectivity and estimate resource usage. The layout is partitioned into Global routing cells (G-Cells) using horizontal and vertical grids, forming a grid graph $G(V, E)$ where vertices represent G-Cells and edges connect adjacent ones.

Based on routing dimensionality, GR can be classified into 2D [14]–[17] and 3D [18]–[21]. In 2D GR, routing is restricted to horizontal and vertical directions and requires subsequent layer assignment, whereas 3D GR spans all three spatial dimensions and determines wire layers during routing. While 3D routing can achieve more efficient resource utilization, 2D routing is faster and often sufficient in practice.

Detailed routing refines global routing guides to produce exact paths at track and via level [22]–[25]. It must satisfy design rules while optimizing wirelength, timing, and efficiency.

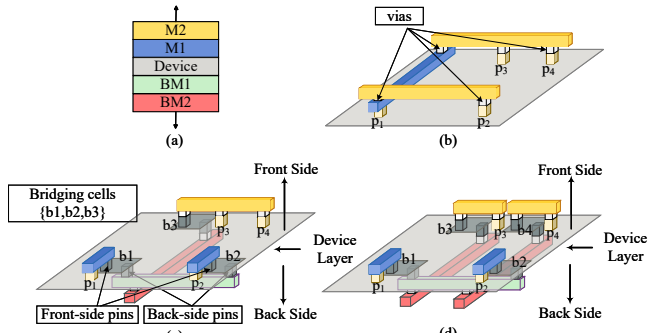


Fig. 2 Routing solution of a net: (a) A double-sided layer structure. (b) A single-sided net on the front-side metal layers. (c) A double-sided net with 2 pins using bridging cells. (d) A double-sided net with all pins using bridging cells.

Typical methods are search-based algorithms, such as maze routing, combined with iterative repair and local optimization, to ensure connectivity and legality.

B. Timing Model

Elmore delay model [26] is adopted to compute net delay. Each net contains one source and multiple sinks. To facilitate computation, multi-pin nets are decomposed into multiple 2-pin subnets and abstracted as RC topologies. The total net delay is obtained by accumulating the delay of each segment, which serves as a key metric for evaluating the timing performance of the routing results [27].

For an segment s in a net, the delay is computed as:

$$d(s) = R(s) \cdot \big(C(s)/2 + C_{\text{down}}(s)\big), \tag{1}$$

where $R(s)$, $C(s)$ denotes resistance and capacitance, respectively and $C_{\text{down}}(s)$ denotes the downstream capacitance accumulated from all downstream nodes connected to the segment.

The total delay of a path P from a sink to the source is the sum of the delays of all segments along the path:

$$d(p) = \sum_{s \in S(P)} d(s), \tag{2}$$

where $S(P)$ is the set of segments on path P.

The total delay of a multi-pin net n is the sum of the delays of all paths in the net, which computed as follows:

$$d(n) = \sum_{p \in P(n)} d(p), \tag{3}$$

where $P(n)$ represents the set of all paths in net n.

C. Double-sided Signal Routing

Fig. 2(a) shows a double-sided routing structure with FS layers M1–M2 and BS layers BM1–BM2, where bridging cells in the device layer connect the two sides through pins on M1 and BM1, and all cross-side signals must pass through them. Both FS upper layers and BS layers, with their lower resistance, effectively improve timing. Fig. 2(b) shows a single-sided routing solution for a 4-pin net, while Figs. 2(c–d) show two double-sided alternatives. In Fig. 2(d), all pins are assigned to the backside, requiring multiple bridging cells. Without precise guidance on the necessity of bridging cells, indiscriminate insertion may cause legalization issues (e.g., overlaps with neighboring standard cells) and increase wirelength or parasitic

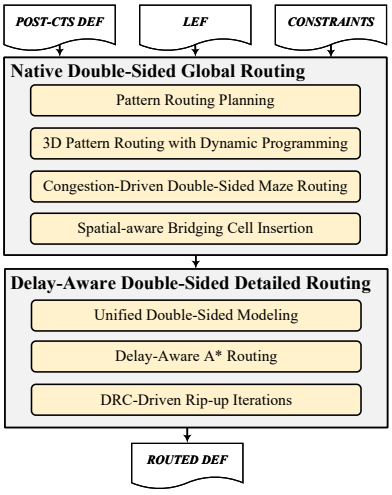


Fig. 3 Overview of DSR framework.

resistance. In contrast, Fig. 2(c) routes only widely separated pins to the backside, thereby reducing the number of bridging cells while still leveraging low-resistance backside layers to preserve timing and mitigate layout risks.

We aim to develop a systematic approach for efficient double-sided routing. Assumptions: (1) the FS consists of m metal layers $F = \{M_1, \ldots, M_m\}$ and the BS consists of n metal layers $B = \{BM_1, \ldots, BM_n\}$; (2) each metal layer has distinct resistance and capacitance values, with upper FS layers and all BS layers offering better timing performance; (3) BSPDN is built on B while the clock network is built on F; (4) all IO pins are on BM_n [28]. We define the native double-sided signal routing problem as follows:

Problem 1 (Native double-sided signal routing): Given a post-CTS DEF design, technology and standard-cell LEFs including back-side resources, and a constraint file, construct routing solutions that assign paths, metal layers, and bridging cells for each net. The output is a fully routed DEF. Routing must satisfy connectivity, design rules, bridging cell legalization, and resource constraints. The objectives are to minimize wirelength, optimize critical-path timing by fully utilizing double-sided metal resources, and eliminate DRVs.

III. DSR APPROACH FRAMEWORK

A. Overview

The overall flow of our proposed approach is shown in Fig. 3. GR generates routing guides that balance congestion, constrain the DR search space to mitigate potential DRVs, and provide timing-aware guidance for critical nets, enabling faster and more efficient routing. Building on this framework, the proposed approach comprises two stages: native double-sided global routing and delay-aware double-sided detailed routing. In the global stage, a 3D delay model guides pattern routing in a unified routing space, where bridging cells are pre-inserted, congestion is resolved through double-sided maze routing. The detailed stage then refines the solution in a delay-driven manner, producing fine-grained routes while repairing DRVs. The details of each stage are presented in the following sections.

B. Native Double-sided Global Routing

In the double-sided global routing stage, the router should leverage low-resistance metal layers to improve timing performance and alleviate congestion, while simultaneously minimizing wirelength and legalizing the insertion of bridging cells. However, 2D routing is insufficient to address such complex and dynamic optimization requirements. To this end, we propose a 3D routing framework that systematically incorporates these objectives.

1) Pattern Routing Planning: This stage is designed to guide subsequent 3D pattern routing and to prepare for accurate bridging-cell capacity estimation. The router generates the Steiner trees for multi-pin nets using Flute [29] and decomposes them into two-pin subnets. Subnets are sorted by bounding box size, prioritizing long critical connections to maintain global routing consistency. Each G-Cell is modeled in detail, storing blocked-track and fixed-macro information for subsequent bridging cell insertion decisions. To control bridging cell insertion, the available capacity of a G-Cell v on the device layer is defined as

$$\text{Cap}(v) = \frac{\text{Area}_G - \text{Area}_c(v)}{\text{Area}_b} \tag{4}$$

where Area_G, $\text{Area}_c(v)$ and Area_b denote the G-Cell area, the total cell and macro area in v, and the area of a bridging cell, respectively. This estimates the number of bridging cells that can be accommodated in the remaining space of the G-Cell.

2) 3D Pattern Routing with Dynamic Programming: After decomposing nets into two-pin subnets in order, a dynamic programming algorithm is applied to each subnet to generate a set of candidate routing paths with simultaneous pattern routing and layer assignment. To achieve better timing performance, each candidate path is then evaluated using the Elmore delay model introduced in Section II-B. The final routing solution is selected from this candidate set, achieving a balanced trade-off among physical feasibility, timing performance, and resource utilization. Let a net node be N_i with a set of child nodes $\{N_{c(1)}, \ldots, N_{c(k)}\}$, and let the total number of metal layers be L. The minimum routing cost of N_i as root on layer l is denoted as $\text{DP}(N_i, l)$, which can be recursively computed as

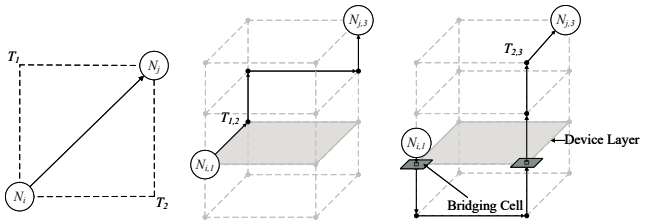
$$\text{DP}(N_i, l) = \\ min_{l_1,\ldots,l_k \in [1,L]} \left\{ \sum_{j=1}^{k} \text{Path}(N_i, N_{c(j)}; l) + \sum_{j=1}^{k} \text{DP}(N_{c(j)}, l_j) \right\}, \tag{5}$$

where $\text{Path}(N_i, N_{c(j)}; l)$ represents the cost of connecting root N_i to child $N_{c(j)}$ on layer l, including wire segment cost, via cost, and possible bridging cell cost. Specifically, the cost of wire segment is

$$c_w(N_i, N_j) = wl(N_i, N_j) + cg(N_i, N_j) + dl(N_i, N_j), \tag{6}$$

where $wl(N_i, N_j)$ is the physical wirelength, $cg(N_i, N_j)$ is the congestion cost, and $dl(N_i, N_j)$ denotes the segment delay, approximated using a simple RC accumulation model:

$$dl(N_i, N_j) = \sum_{s \in \text{Seg}(N_i, N_j)} \alpha \cdot R_s \cdot C_s, \tag{7}$$



(a) (b) (c)

Fig. 4 Comparison of routing strategies

where $\text{Seg}(N_i, N_j)$ representing the set of wire segments between nodes N_i and N_j, R_s and C_s the resistance and capacitance of segment s, respectively, and α a scaling factor. This allows efficient estimation of segment delays by accumulating the RC contributions of all segments.

The cost associated with bridging cells is defined as

$$c_b(N_j) = \text{cong}(N_j) + dl_b + \text{Area}_b \cdot e^{-k \cdot \text{Cap}(N_j)}, \quad (8)$$

where $\text{cong}(N_j)$ denotes congestion on the device layer, dl_b is the bridging cell delay cost, Area_b is the cell area, $\text{Cap}(N_j)$ is the bridging cell capacity in N_j, and k is a tunable parameter. To enable flexible adjustment of bridging cells during pattern routing and improve routing efficiency, they are **pre-inserted** to reserve resources, acting as a constraint for path search without committing to final placement.

We define the base via cost along a path as

$$c_{via}(N_j, l, l') = cg(N_j, l, l') + dl(N_j, l, l') + uvc \cdot n_v, \quad (9)$$

where $cg(N_j, l, l')$ and $dl(N_j, l, l')$ are the total congestion and delay costs from layer l to l', uvc is the unit via cost, and n_v is the number of vias along the path.

Finally, the via cost including potential bridging cell contribution is expressed as

$$c_v(N_j, l, l') = \begin{cases} c_{via}(N_j, l, l') + c_b(N_j), & l \in \text{BS or } l' \in \text{BS}, \\ c_{via}(N_j, l, l'), & \text{otherwise.} \end{cases} \quad (10)$$

For a two-pin net $N_i \to N_j$, the minimum routing cost along a 3D L-shaped path is

$$\text{Path}(N_i, N_j; l) = \min_{\substack{l_1,\ldots,l_k \in [1,L] \\ x=1,2}} \Big\{ c_w(N_i, T_x) + c_v(N_i, l, l_{T_x}) + c_w(T_x, N_j) + c_v(N_j, l_{T_x}, l_j) + \text{DP}(N_j, l_j) \Big\}, \quad (11)$$

where T_x denotes the turning point and x represents the two possible L-shaped bending choices. The algorithm recursively computes minimum costs for child nodes and enumerates 3D L-shaped paths to obtain the optimal routing for the net.

Fig. 4(a) shows the 2D structure of a two-pin net, and Fig. 4(b) shows the the 3D structure of one possible path. In the example, T_1 is chosen as the turning point. Fig. 4(c) depicts another solution involving back-side routing, where T_2 serves as the turning point. In this case, the feasibility of the bridging cell is checked and then pre-inserted in the device layer to transfer signals to the metal layers of the backside, and the path between N_i and T_2 is routed on the back side.

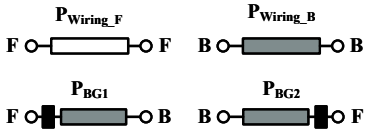


Fig. 5 Patterns of wire segments for signal routing. **F** denotes the front-side, and **B** denotes the back-side. The left and right ends indicate the source and sink positions of the signal, respectively.

3) Congestion-Driven Double-Sided Maze Routing: Following pattern routing, the router performs multiple Rip-up and Reroute (RRR) iterations on unrouted or violating nets to alleviate congestion and restore connectivity. Bridging cells pre-inserted during pattern routing remain fixed but are referenced when constructing the 3D congestion graph $G_c(V, E)$ to identify G-Cells with BS access. A multi-source multi-sink Dijkstra algorithm [30] expands paths from all pin access points, while a union–find structure dynamically merges connected components. Routing completes once all pins belong to a single component, after which resource usage is updated by backtracking. Vias not only affect resource usage but also play a critical role in timing optimization, and the cost are therefore defined as follows:

$$Cost(e) = CongestionCost(e) + ViaPenalty(e), \quad (12)$$

where $CongestionCost(e)$ is dynamically estimated from historical routing states, and $ViaPenalty(e)$ suppresses excessive via use. After maze routing, the pre-inserted bridging cells are committed to the routing paths, ensuring consistency and manufacturability of the double-sided solution.

4) Spatial-aware Bridging Cell Insertion: Global routing impacts systematic metrics such as timing, wirelength, and DRVs, while constraining the solution space for detailed routing. Therefore, bridging cell positions are finalized at the end of global routing, enabling detailed routing to focus on DRV repair and timing optimization, improving efficiency and convergence. When a pin needs to be routed on the BS, the algorithm queries a lookup table built during the pattern routing planning stage, which records each G-Cell's available space. Through this query, it efficiently identifies candidate insertion sites and updates the table to reflect the latest resource usage.

The double-sided design space is defined by the discrete edge patterns and connectivity constraints. Unlike traditional single-sided routing, the edge patterns in our algorithm are designed to adapt both sides and bridging cells. We list four candidate edge patterns in Fig. 5: $\mathbf{P}_{\text{Wiring\_F}}$ for routing on the front side, $\mathbf{P}_{\text{Wiring\_B}}$ for the back side, and since the two pins of a bridging cell are located on different sides, resulting in edge endpoints with distinct side types, $\mathbf{P}_{\text{BG1}}$ for switching from front to back, and $\mathbf{P}_{\text{BG2}}$ for switching from back to front. Bridging cell insertion must satisfy connectivity constraints: adjacent edges share a node with consistent side type. If a node lies on different sides, a bridging cell is inserted to enable side switching and guarantee continuity.

Fig. 6 illustrates a pre-insertion case for pin p_1. When back-side access is required, the algorithm sequentially scans the lookup table row by row. The first row has no available sites, the second provides a candidate site but introduces overlap with a neighboring standard cell, while the third row yields a legal

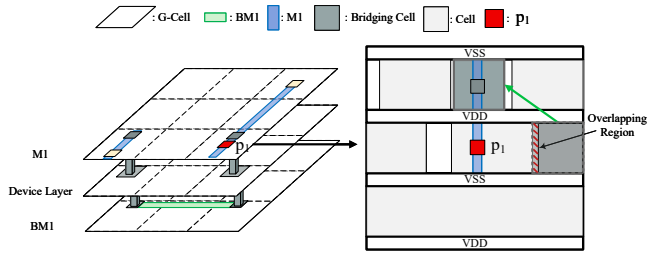


Fig. 6 Pre-insertion of bridging cells within a G-Cell.

site for pre-insertion. If no valid site is found across all rows, pin p_1 must instead choose an alternative routing path that avoids back-side access. After global routing, the pre-inserted bridging cells are finalized by instantiating them in the DEF and updating the associated nets, ensuring that they are seamlessly incorporated into the routed topology through their pins.

C. Delay-Aware Double-Sided Detailed Routing

The proposed DSR framework performs double-sided detailed routing through a structured three-stage process as shown in Algorithm 1: unified modeling, delay-aware A* search, and DRC-driven rip-up. Each stage is designed with a clear objective: improving runtime efficiency, optimizing timing, and ensuring design-rule compliance.

1) Unified Double-Sided Modeling: Distinguish from conventional single-sided routing, the router first constructs a unified graph representing both FS and BS metal layers. Bridging cells provide access to BS resources. Design rules, congestion maps, and routing guides are then loaded to define legal routing regions, and initial candidate regions are assigned for each net. Unified modeling allows each net to be routed with a single search on both sides, thereby accelerating runtime and preventing layer-assignment conflicts (Lines 1–2).

2) Delay-Aware A Routing:* To optimize timing-critical paths, each net is routed using a delay-aware A* search. Candidate nodes are iteratively expanded, accumulating resistance and capacitance to estimate Elmore delay. The composite cost function $c = f(\text{DRC}, \text{hist}, \text{via}, \text{turn}, \text{area}, \text{delay})$ balances design-rule violation penalties, congestion history, via, turning costs, area congestion, and estimated delay. By prioritizing nodes with lower cost, the router favors timing-friendly and legal paths while still satisfying design rules. Once a feasible path is found, resources along the path are reserved to prevent conflicts with subsequent nets (Lines 3–11).

3) DRC-Driven Rip-up Iterations: Following the DRC-driven RRR framework of TritonRoute [22], the router identifies nets or regions with remaining violations or congestion and selectively rips up their paths. Delay-aware A* search is then re-applied to reconstruct legal, timing-optimized paths. This iterative rip-up and reroute process continues until all design rules are satisfied and congestion is mitigated. This stage ensures correctness and manufacturability while maintaining timing performance (Lines 12–15).

IV. Experiment

Experiments are conducted on a Linux workstation with a 3.5GHz Intel Xeon processor and 128GB of memory. The technology platform is based on the ASAP7 PDK [31], and

Algorithm 1 Delay-Aware Double-Sided Detailed Routing

Input: LEF, DEF, Global routing guide
Output: Routed DEF
1: Initialize double-sided routing graph and bridging cells
2: Load design rules, congestion map, and routing guides
3: **for all** net in Nets **do**
4: Initialize path search
5: **while** path not found **do**
6: Compute cumulative R/C and Elmore delay
7: Expand node with
8: $c = f(\text{DRC}, \text{hist}, \text{via}, \text{turn}, \text{area}, \text{delay})$
9: **end while**
10: Reserve routing resources
11: **end for**
12: **while** violations exist or congestion excessive **do**
13: Identify affected nets/regions
14: Rip-up affected paths
15: Re-run delay-aware A* routing
16: **end while**
17: **return** Routed nets

the back-side metal layers as well as the unit resistance and capacitance parameters of bridging cell are adopted from [32], with detailed values listed in Table I. The benchmark designs are derived from OpenROAD [33], and use its backend flow to generate benchmarks. The statistics of the benchmarks are listed in Table II. The framework proposed in this work is implemented in C++.

A. Technology Settings

For delay estimation, we follow the OpenROAD convention: front-side wires use M3 unit resistance and capacitance, while back-side wires are estimated using the actual BM1–BM3 parameters [10]. Each bridging cell is modeled as an nTSV [4] of $0.27\ \mu\text{m} \times 0.27\ \mu\text{m}$, which is aligned to the other standard cells in the layout. To ensure manufacturability of a bridging cell's back-side pin, signal routing on BM1 is prohibited [5]. The resistance and capacitance of one bridging cell is $0.020\ \text{k}\Omega$ and 0.004 fF [10].

B. Comparison of Signal Routing

To compare single-sided and double-sided routing and evaluate our proposed double-sided approach, we used a commercial

TABLE I Layer resistances and capacitances [31], [32]

Layer	Unit Res. ($k\Omega/\mu m$)	Unit Cap. ($f\text{F}/\mu m$)
M1	0.13889	0.11368
M2	0.024222	0.13426
M3	0.024222	0.12918
M4	0.016778	0.11396
M5	0.014667	0.13323
M6	0.010371	0.11575
M7	0.009672	0.13293
M8	0.007431	0.11822
M9	0.006874	0.13497
BM1–BM3	0.000384	0.116264

TABLE II The statistics of benchmarks [33]

ID	Design	#Nets	#Pins	#Cells
C1	gcd	451	1340	1371
C2	uart	700	2264	1653
C3	aes	17420	55999	17640
C4	jpeg	76816	212883	170410
C5	ethmac	60758	199822	143258
C6	aes-mbff	17479	56114	36205
C7	mock-alu	14307	41298	24985
C8	jpeg-lvt	65214	122136	149901
C9	ethmac_lvt	60719	202039	63174

TABLE III Comparison of Single-sided and Double-sided Routing Approaches. "WL", "TD", "MD", "DRV", and "BG" denote wirelength (μm), total delay (ps), maximum delay (ps), number of DRVs, and number of bridging cells, respectively.

Design	Innovus (Single-sided)				Double-Sided with Netlist Planning [4]					DSR				
	WL	TD	MD	DRV	WL	TD	MD	DRV	BG	WL	TD	MD	DRV	BG
C1	1174	7325	379	0	**989**	6792	442	0	60	991	**4424**	**369**	0	12
C2	1489	7966	473	0	1333	6265	272	0	8	**1319**	**4680**	**250**	0	6
C3	76867	1504805	8832	0	74646	873066	7231	0	166	**74374**	**759188**	**5908**	0	204
C4	292333	15903436	75048	0	**260543**	7027859	69212	0	667	267736	**6102486**	**55370**	0	540
C5	472794	33969374	51624	0	436399	9952697	36680	0	470	**431946**	**8654528**	**28779**	0	647
C6	77011	1552509	10220	0	**76276**	930908	7919	0	111	76330	**809483**	**6335**	0	161
C7	53260	537564	16662	0	39867	476544	5560	0	224	**39836**	**414382**	**4448**	0	28
C8	286873	16453458	70030	0	**277647**	7094444	48513	0	1058	280217	**6169038**	**38810**	0	1172
C9	509862	34055750	38803	0	**429103**	9562644	38891	0	2002	437892	**8315342**	**31113**	0	984
Ratio	1.12	2.42	1.77	-	1.00	1.21	1.22	-	1.27	**1.00**	**1.00**	**1.00**	-	**1.00**

TABLE IV Ablation study on the effectiveness of double-sided detailed routing. "WL", "TD", "MD", "DRV", and "RT" denote wirelength (μm), total delay (ps), maximum delay (ps), number of DRVs, and runtime, respectively.

Design	TritonRoute					DSR w/o DA					DSR				
	WL	TD	MD	DRV	RT	WL	TD	MD	DRV	RT	WL	TD	MD	DRV	RT
C1	**989**	6630	438	0	18.18	991	**4424**	**369**	0	12.63	991	**4424**	**369**	0	12.95
C2	1333	5961	226	0	16.32	1327	4688	**158**	0	14.23	**1319**	**4680**	250	0	14.60
C3	74610	944980	8978	0	904.19	**74113**	**758700**	8923	0	428.50	74374	759188	**5908**	0	432.21
C4	261487	6954725	53735	8	3463.89	**252721**	6987012	60677	0	2908.17	267736	**6102486**	**55370**	0	2968.95
C5	434696	11107419	69784	52	7864.82	437620	12854540	52653	0	4093.45	**431946**	**8654528**	**28779**	0	4173.86
C6	76501	955262	6585	8	1083.83	**76271**	917677	**6148**	0	417.05	76330	**809483**	6335	0	421.92
C7	39904	458976	4494	0	570.03	**39601**	450743	**3914**	0	203.12	39836	**414382**	4448	0	205.81
C8	287670	7093192	48513	7	4085.67	**279992**	**5619051**	**36331**	0	2480.36	280217	6169038	38810	0	2524.42
C9	449088	9610050	46413	118	5535.74	448341	9774532	33345	0	3180.54	**437892**	**8315342**	**31113**	0	3249.08
Ratio	1.01	1.22	1.33	-	1.64	1.00	1.15	1.06	-	0.98	1.00	1.00	1.00	-	1.00

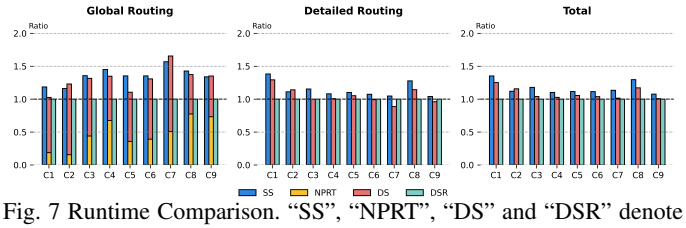


Fig. 7 Runtime Comparison. "SS", "NPRT", "DS" and "DSR" denote single-sided approach, netlist planning runtime, double-sided approach [4] and our approach DSR, respectively.

place and route tool, Cadence Innovus [34], to produce a single-sided routing solution for each benchmark. For the single-sided benchmarks, we add two additional top metal layers, M10 and M11 [4], assigning them the same RC values as the BS metal layers in Table I. The corresponding double-sided routing is produced following the procedure in work [4]. Table III summarizes the comparison. The results show that double-sided routing significantly improves wirelength, timing, and routing efficiency over single-sided routing. Compared with the alternative double-sided approach [4], DSR reduces total delay and max delay by approximately 17.4% and 18.1%, respectively, and requires about 21.3% fewer bridging cells, while maintaining comparable wirelength.

Fig. 7 further reports runtime. DSR outperforming both single-sided and netlist-planning double-sided routing. This improvement is mainly due to the native double-sided routing framework, which systematically considers optimization objectives within a unified double-sided routing space, avoiding the overhead of netlist generation and subnet partitioning required by netlist-planning approaches.

C. Ablation Study

To evaluate the effectiveness of the proposed DSR double-sided detailed routing, three schemes are considered: 1) *TritonRoute*:Under the guidance of DSR's native double-sided global routing, TritonRoute performs detailed routing separately on each side; 2) *DSR w/o DA*: double-sided detailed routing without delay-aware strategy; 3) *DSR*: double-sided detailed routing with delay-aware strategy. Table IV shows that TritonRoute exhibits significant DRVs in the double-sided context. As a single-sided router, it only guarantees connectivity and legality on one side, lacking unified modeling and constraints when routing separately, causing overlaps and spacing errors at bridging interfaces. In contrast, DSR performs unified modeling and search across both sides, thereby eliminating such issues, achieving zero DRVs, and reducing runtime by up to 39.0%, while also improving total and max delays by 18.0% and 24.8%, respectively, compared with TritonRoute.

Although the global routing guide already achieves satisfactory timing, integrating the delay-aware A* search in DSR reduces the average total delay by 13.1% and the maximum delay by 5.7% compared to DSR without delay awareness. With minimal impact on wirelength, the combined strategy of unified double-sided modeling and delay-aware search yields the best overall performance, ensuring zero DRVs, accelerating routing, and optimizing critical-path timing.

V. Conclusion

In this work, we propose a native double-sided signal routing approach for the first time, aiming to efficiently explore double-sided routing resources to address the complex systemic requirements of signal routing. We construct a unified model of double-sided resources and develop a native double-sided global routing algorithm that performs dynamic path optimization in 3D space under delay models. By extending the double-sided routing graph and introducing delay-aware mechanisms, we further propose a double-sided detailed router, which enhances both performance and efficiency under the guidance of global routing result. Experimental results demonstrate the effectiveness of the proposed approach in addressing the systematic requirements of signal routing.

REFERENCES

[1] E. Beyne, A. Jourdain, and G. Beyer, "Nano-through silicon vias (ntsv) for backside power delivery networks (bspdn)," in *2023 IEEE Symposium on VLSI Technology and Circuits*, 2023, pp. 1–2.

[2] A. Veloso, B. Vermeersch, R. Chen, P. Matagne, M. G. Bardon, G. Eneman *et al.*, "Backside power delivery: game changer and key enabler of advanced logic scaling and new stco opportunities," in *2023 International Electron Devices Meeting (IEDM)*, 2023, pp. 1–4.

[3] D. Prasad, S. T. Nibhanupudi, S. Das, O. Zografos, B. Chehab *et al.*, "Buried power rails and back-side power grids: Arm® cpu power delivery network design beyond 5nm," in *2019 IEEE International Electron Devices Meeting (IEDM)*, 2019, pp. 19.1.1–19.1.4.

[4] T.-C. Lin, F.-Y. Hsu, W.-K. Mak, and T.-C. Wang, "An effective netlist planning approach for double-sided signal routing," in *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2024, pp. 288–293.

[5] F.-Y. Hsu, T.-C. Lin, W.-K. Mak, and T.-C. Wang, "A bounding box-based net partitioning method for double-sided routing," in *Proceedings of the Great Lakes Symposium on VLSI 2024*, 2024, pp. 397–402.

[6] N. E. Bethur, P. Vanna-Iampikul, O. Zografos, L. Zhu, G. Sisto, D. Milojevic *et al.*, "Gnn-assisted back-side clock routing methodology for advance technologies," in *2024 61st ACM/IEEE Design Automation Conference (DAC)*, 2024, pp. 1–6.

[7] N. E. Bethur, "A methodology for back-side clock delivery network design compatible with commercial eda flows," Master's thesis, Georgia Institute of Technology, 2023.

[8] M. M. S. Aly, T. F. Wu, A. Bartolo, Y. H. Malviya, W. Hwang, G. Hills *et al.*, "The n3xt approach to energy-efficient abundant-data computing," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 19–48, 2018.

[9] H. Lu, Y. Ge, X. Jiang, J. Sun, W. Peng, R. Guo *et al.*, "First experimental demonstration of self-aligned flip fet (ffet): A breakthrough stacked transistor technology with 2.5 t design, dual-side active and interconnects," in *2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, 2024, pp. 1–2.

[10] X. Jiang, H. Lu, Y. Zhao, J. Wang, Z. Guo, H. Wu *et al.*, "A systematic approach for multi-objective double-side clock tree synthesis," in *2025 62nd ACM/IEEE Design Automation Conference (DAC)*, 2025, pp. 1–7.

[11] M. Shamanna, E. Abuayob, G. Aenuganti, C. Alvares, J. Antony, A. Bahudhanam *et al.*, "E-core implementation in intel 4 with powervia (backside power) technology," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, 2023, pp. 1–2.

[12] "Tsmc a16 technology," https://www.tsmc.com/english/dedicatedFoundry/technology/logic/lA16.

[13] C.-P. Tsai, F.-Y. Hsu, W.-K. Mak, and T.-C. Wang, "An effective eco methodology for reducing back-side design rule violations in double-sided signal routing," in *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, 2024, pp. 1–9.

[14] W. Li, R. Liang, A. Agnesina, H. Yang, C.-T. Ho, A. Rajaram *et al.*, "Dgr: Differentiable global router," in *2024 61st ACM/IEEE Design Automation Conference (DAC)*, 2024, pp. 1–6.

[15] M. Cho, K. Lu, K. Yuan, and D. Z. Pan, "Boxrouter 2.0: Architecture and implementation of a hybrid and robust global router," in *2007 IEEE/ACM International Conference on Computer-Aided Design*, 2007, pp. 503–508.

[16] J. Hu, J. A. Roy, and I. L. Markov, "Sidewinder: a scalable ilp-based router," in *Proceedings of the 2008 International Workshop on System Level Interconnect Prediction*, 2008, pp. 73–80.

[17] J. He, U. Agarwal, Y. Yang, R. Manohar, and K. Pingali, "Sproute 2.0: A detailed-routability-driven deterministic parallel global router with soft capacity," in *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2022, pp. 586–591.

[18] J. Liu and E. F. Young, "Edge: Efficient dag-based global routing engine," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.

[19] J. Liu, C.-W. Pui, F. Wang, and E. F. Young, "Cugr: Detailed-routability-driven 3d global routing with probabilistic resource model," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[20] S. Lin and M. D. Wong, "Superfast full-scale cpu-accelerated global routing," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, 2022, pp. 1–8.

[21] S. Liu, Y. Pu, P. Liao, H. Wu, R. Zhang, Z. Chen *et al.*, "Fastgr: Global routing on cpu–gpu with heterogeneous task graph scheduler," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 7, pp. 2317–2330, 2022.

[22] A. B. Kahng, L. Wang, and B. Xu, "Tritonroute: The open-source detailed router," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 3, pp. 547–559, 2020.

[23] H. Li, G. Chen, B. Jiang, J. Chen, and E. F. Young, "Dr. cu 2.0: A scalable detailed routing framework with correct-by-construction design rule satisfaction," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2019, pp. 1–7.

[24] S. M. Goncalves, L. S. d. Rosa Jr, and F. S. Marques, "Smartdr: Algorithms and techniques for fast detailed routing with good design rule handling," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 26, no. 2, pp. 1–38, 2020.

[25] Z. Zhuang, G. Liu, T.-Y. Ho, B. Yu, and W. Guo, "Trader: A practical track-assignment-based detailed router," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2022, pp. 766–771.

[26] J. Hu, G. Schaeffer, and V. Garg, "Tau 2015 contest on incremental timing analysis," in *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2015, pp. 882–889.

[27] G. Liu, X. Zhang, W. Guo, X. Huang, W.-H. Liu, K.-Y. Chao *et al.*, "Timing-aware layer assignment for advanced process technologies considering via pillars," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 6, pp. 1957–1970, 2022.

[28] E. Beyne and J. Ryckaert, "An integrated circuit chip with power delivery network on the backside of the chip," Patent EP3 324 436A1, 2017.

[29] C. Chu and Y.-C. Wong, "Flute: Fast lookup table based rectilinear steiner minimal tree algorithm for vlsi design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, pp. 70–83, 2007.

[30] Y.-J. Chang, Y.-T. Lee, J.-R. Gao, P.-C. Wu, and T.-C. Wang, "Nthuroute 2.0: a robust global router for modern designs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 12, pp. 1931–1944, 2010.

[31] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline *et al.*, "Asap7: A 7-nm finfet predictive process design kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016.

[32] R. Chen, G. Sisto, A. Jourdain, G. Hiblot, M. Stucchi, N. Kakarla *et al.*, "Design and optimization of sram macro and logic using backside interconnects at 2nm node," in *2021 IEEE International Electron Devices Meeting (IEDM)*, 2021, pp. 22.4.1–22.4.4.

[33] OpenROAD Project, "Openroad," https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts.

[34] "Innovus implementation system, 21.13 ed." https://www.cadence.com/.