JRouter: A Multi-Terminal Hierarchical Length-Matching Router under Planar Manhattan Routing Model for RSFQ Circuits

Xinda Chen
ShanghaiTech University
Shanghai, China

Rongliang Fu
The Chinese University of Hong Kong
Hong Kong, China

Junying Huang*
Institute of Computing Technology, CAS
Beijing, China

Huawei Cao
Institute of Computing Technology, CAS
Beijing, China

Zhimin Zhang
Institute of Computing Technology, CAS
Beijing, China

Xiaochun Ye
Institute of Computing Technology, CAS
Beijing, China

Tsung-Yi Ho
The Chinese University of Hong Kong
Hong Kong, China

Dongrui Fan
University of Chinese Academy of Sciences
Beijing, China

ABSTRACT

Superconducting rapid single-flux-quantum (RSFQ) logic has shown great potential for high-energy-efficient computing systems. To ensure correct operations at ultra-high frequencies, it is necessary to incorporate length-matching constraints into the routing problem. Existing routing algorithms, however, can only address 2-pin connections or support the conventional horizontal/vertical routing model, which substantially limits the optimization space for routing solutions. This paper presents JRouter, an RSFQ router that considers the two-layer planar Manhattan routing model while simultaneously coping with splitter (SPL) placement and length-matching multi-terminal routing. JRouter contains a track-assignment-based initial routing that minimizes the initial routing width while avoiding conflicts in the horizontal constraint graph. Moreover, JRouter implements an SPL-tree-based hierarchical routing with an iterative maximum-flow-based formulation to insert the detours for multi-terminal routing. A routing region extension algorithm is also developed to insert the detours for unsatisfied connections. According to the experimental results, JRouter achieves an average routing width reduction of 35.71% and 22.46% on a 16-bit RSFQ Sklansky adder compared to Kito's and Kou's routing algorithms. For randomly generated benchmarks, JRouter reduces the routing width by an average of 38.77%, 38.20%, 21.65%, and 7.01% compared to Kito's, Kou's, and two of Yan's routing algorithms, respectively, while maintaining reasonable runtime.

CCS CONCEPTS

• **Hardware → Wire routing**; **Emerging technologies**.

KEYWORDS

Superconducting Logic, RSFQ, Routing, Length-matching

ACM Reference Format:
Xinda Chen, Rongliang Fu, Junying Huang, Huawei Cao, Zhimin Zhang, Xiaochun Ye, Tsung-Yi Ho, and Dongrui Fan. 2023. JRouter: A Multi-Terminal Hierarchical Length-Matching Router under Planar Manhattan Routing Model for RSFQ Circuits. In *Proceedings of the Great Lakes Symposium on VLSI 2023 (GLSVLSI '23), June 5–7, 2023, Knoxville, TN, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3583781.3590267

*Corresponding author: huangjunying@ict.ac.cn

GLSVLSI '23, June 5–7, 2023, Knoxville, TN, USA

ACM ISBN 979-8-4007-0125-2/23/06.
https://doi.org/10.1145/3583781.3590267

1 INTRODUCTION

Currently, we are running out of an effective option to propel the computer system's performance further while maintaining its power budget in the post-Moore era. Innovative technologies like quantum, neuromorphic, approximate, and stochastic computing may solve these challenges. Among these candidates, Josephson Junction-based superconducting single-flux-quantum (SFQ) logic family is an up-and-coming solution with potential benefits including ultra-fast switching speed ($\sim$ 1 ps) and ultra-low switching energy ($\sim 10^{-19}$ J/bit), which is six orders of magnitude smaller than semiconductor transistors [4, 8]. Besides, as a member of superconducting SFQ logic families, the rapid single flux quantum (RSFQ)-based T-flip flop has been demonstrated that can operate at up to 770 GHz at 4.2 K [1]. This technology has the potential to significantly improve device clock frequencies (and thus performance) by orders of magnitude [7].

RSFQ logic employs quantized voltage pulses as digital data generation signals, enabling fast switching. However, for correct operations at increasing clock frequencies, it becomes crucial to adjust timing to meet the timing requirements of each logic cell. In RSFQ circuits, timing adjustment can be achieved by inserting delay elements such as Josephson Transmission Lines or by extending the length of Passive Transmission Lines (PTLs). Since PTLs are passive routing cells with timing-variability independent of the delay time, they can increase operation margin and are intensively used as interconnects in RSFQ circuits. The requirement for timing adjustment is thus converted to length-matching constraints in the routing problem. In other words, extension lengths of PTLs can be

Table 1: Comparison of RSFQ length-matching routing.

Routing algorithm	Multi-terminal routing	Planar Manhattan routing model	SPL-aware routing
Kito's [5]	No	No	No
Kou's [6]	Yes	No	Yes
Yan's [9]	No	Yes	No
JRouter	**Yes**	**Yes**	**Yes**

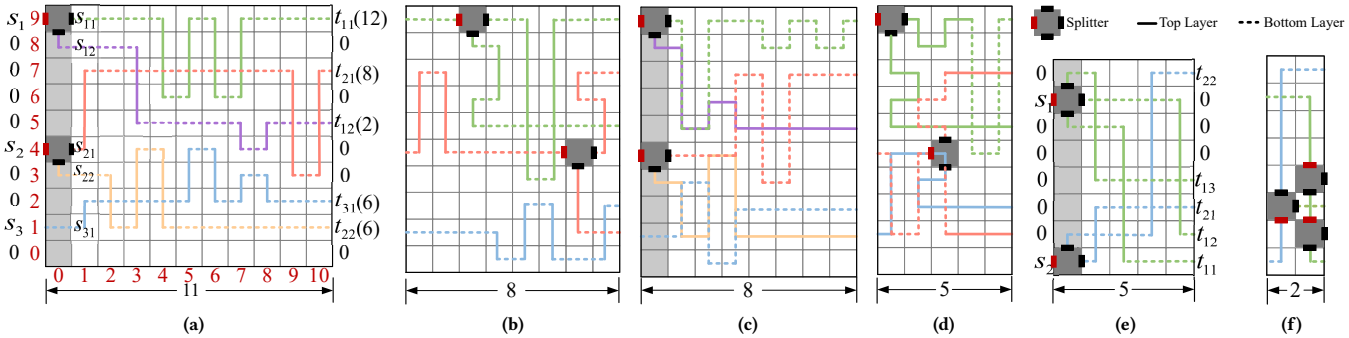

Figure 1: Solutions of an example of the length-matching routing problem using (a) Kito's routing algorithm [5], (b) Kou's routing algorithm [6], (c) Yan's routing algorithm [9], and (d) Ours. A comparison between (e) SPL pre-allocation and (f) SPL reallocation. Numbers in red indicate the row and column positions, the leftmost column in gray is the SPL pre-allocated region, and grids in white are the PTL region.

inserted between two adjacent logic-gate columns for all connections to meet the timing adjustment. Moreover, unlike CMOS-based gates, most RSFQ logic gates can only drive one fan-out. To overcome this limitation, a specific cell called the splitter (SPL) has been developed to split a signal and enable multiple fan-outs.

Due to considerable clock trees distributed in RSFQ circuits, routing resources are limited, causing high routing complexity. Numerous related work has been proposed to solve this problem. Kito et al. first proposed an integer linear programming-based method to minimize the routing width while considering length-matching constraints [5]. However, this method heavily relies on the two-layer horizontal/vertical (HV) routing model, where one layer can only be used for horizontal wires and the other for vertical wires, significantly limiting the routing efficiency. To further reduce the routing width for a set of 2-pin connections with extension lengths, Yan [9] presented an efficient two-layer planar Manhattan routing model that can assign any wire in any direction on the top and bottom layers, allowing efficient use of limited routing resources in two routing layers. In addition, to mitigate the adverse effect of the impedance mismatch in the routing step, Yan [10] also proposed an efficient via-minimization-oriented routing algorithm. However, in [5, 9, 10], SPLs are pre-placed at the logic-gate column, which may degrade the routability when performing length-matching routing, as they hinder the utilization of limited routing resources. Some work [2, 6] has recognized the importance of simultaneously optimizing SPL placement and routing, thus proposing the SPL-aware multi-terminal routing algorithm. However, they are also limited to the two-layer HV routing model.

This paper proposes JRouter, the first routing algorithm for RSFQ circuits that combines the two-layer planar Manhattan routing model and the multi-terminal routing algorithm, as well as handling SPL placement in the routing. Table 1 compares JRouter with the state-of-the-art length-matching RSFQ routing methods.

In summary, this paper makes the following contributions:

- We propose a track-assignment-based initial routing algorithm that can minimize the initial routing width while avoiding conflicts in assigning PTL segments.
- We design an SPL tree-based hierarchical routing for length-matching to make more efficient use of limited routing resources when inserting detours for multi-terminal routing.
- We propose a routing region extension algorithm to insert the detours onto one extra region for unsatisfied connections.
- JRouter achieves an average routing width reduction of 35.71% and 22.46% on a 16-bit RSFQ Sklansky adder compared to Kito's[5] and Kou's[6] algorithms, and 38.77%, 38.20%, 21.65% and 7.01% on randomly generated benchmarks compared to Kito's[5], Kou's[6], and Yan's [9, 10] algorithms, respectively, with reasonable runtime.

2 PROBLEM FORMULATION AND MOTIVATIONS

2.1 Problem Formulation

The delay of a PTL connection can be roughly proportional to its length. Hence, in a timing-variability-aware RSFQ circuit, precise timing adjustment is necessary for correct operations, which can be achieved by setting length-matching constraints on each PTL connection. These length-matching constraints can be treated as extension lengths on the PTL connections. In length-matching routing, it is assumed that a grid-based rectangular PTL region between two adjacent logic-gate columns is considered. Since the routing height of one PTL region is fixed after the placement of logic-gates, its routing area can be minimized by reducing the routing width.

Given two sets $S = \{s_1, s_2, \cdots, s_i, \cdots, s_m\}$ and $T = \{(t_{11}, e_{11}), \cdots, (t_{1n_1}, e_{1n_1}), \cdots, (t_{ij}, e_{ij}), \cdots, (t_{in_i}, e_{in_i}), \cdots, (t_{mn_m}, e_{mn_m})\}$, the connections are defined as $\{s_1 \rightarrow t_{11}, \cdots, s_1 \rightarrow t_{1n}, \cdots, s_i \rightarrow t_{ij}, \cdots, s_i \rightarrow t_{in_i}, \cdots, s_m \rightarrow t_{mn_m}\}$, where s_i and t_{ij} represent the source and destination positions of the connection, n_i stands for the fan-outs count of s_i, and e_{ij} indicates the required extension length of connection $si \rightarrow t_{ij}$. The objective of the length-matching routing problem is to minimize the routing width such that there are no unsatisfied connections, i.e. each connection satisfies its length-matching constraint. That is, the total routing length of a connection (d_{ij}) should satisfy Equation (1), where w denotes the routing width of the PTL region.

$$d_{ij} = |s_i - t_{ij}| + e_{ij} + w, \tag{1}$$

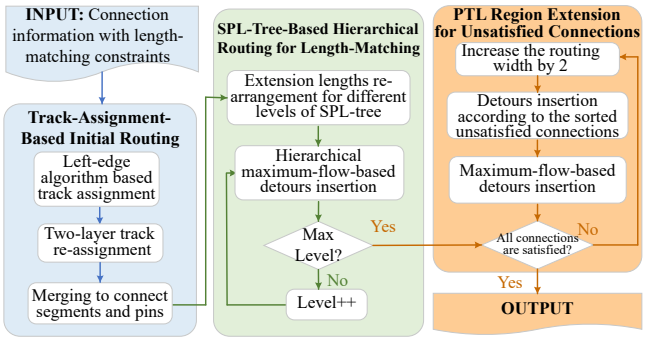

Figure 2: The flow of JRouter.

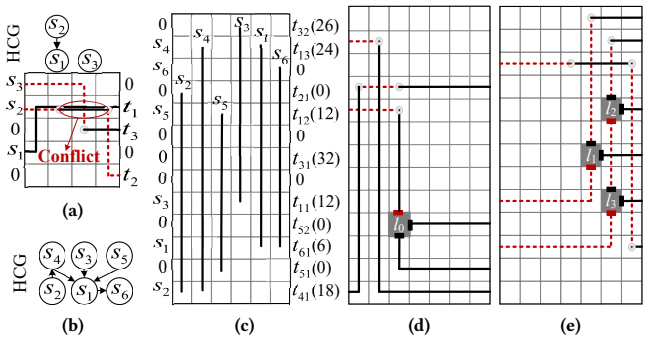

Figure 3: (a) Horizontal conflict with track assignment model [9]. (c) Source and destination pin positions of connections and the initial track assignment. (b) HCG that describes the relative positions of (c). Result of track re-assignment on the (d) top and (e) bottom layer corresponding to (c).

According to the assumptions in [5], the extension lengths must be rounded to even. We assume that there are two kinds of SPLs, SPL-2 and SPL-3, which can generate two and three fan-outs in different directions. We also assume that all SPLs have an area of 1×1. As shown in Fig. 1(a), $s_1 = 9, s_2 = 4, s_3 = 1, (t_{11}, e_{11}) = (9, 12), (t_{12}, e_{12}) = (5, 2), (t_{21}, e_{21}) = (7, 8), (t_{22}, e_{22}) = (1, 6), (t_{31}, e_{31}) = (2, 6)$. Take $s_1 \rightarrow t_{11}$, marked with a green line, as an example. The total routing length from s_1 to t_{11} is 23, since $w = 11$, $|s_1 - t_{11}| = 0$, and $e_{11} = 12$. Hence, 12 units of the extension length must be inserted to satisfy Equation (1).

2.2 Motivations

Fig. 1(a) shows a length-matching routing example using Kito's routing algorithm [5]. The leftmost logic-gate column is used for SPL placement. After SPL pre-allocation, the problem illustrated in Section 2.1 can be converted to matching multiple 2-pin connections. In this example, they are $s_{11} \rightarrow t_{11}, s_{12} \rightarrow t_{12}, s_{21} \rightarrow t_{21}, s_{22} \rightarrow t_{22}$ and $s_{31} \rightarrow t_{31}$ from the second column to the last column of the PTL region, with the resultant routing width of 11.

SPL pre-allocation requires that each 2-pin connection be assigned a separate vertical PTL segment. As shown in Fig. 1(e), five segments are assigned among four tracks of the PTL region since there are five 2-pin connections, and thus five nets, after SPL pre-allocation. However, as illustrated in Fig. 1(f), there are only two nets, and each t_{ij} started from s_i can share one vertical segment if SPLs are reallocated, making the routing width much narrower. Based on this observation, [6] designed a multi-terminal length-matching routing algorithm to consider SPL placement. Fig. 1(b) illustrates its solution for the same problem as in Fig. 1(a). On the other side, considering the connections $s_2 \rightarrow t_{21}$ and $s_2 \rightarrow t_{22}$ indicated in red, six units of extension length are inserted at the front end of the SPL. Therefore, $s_2 \rightarrow t_{21}$ and $s_2 \rightarrow t_{22}$ only require two units and zero units of PTL extension length, saving six units of extension length compared with Fig. 1(a). As a result, the routing width decreases to 8 with SPL reallocation.

However, these two algorithms mentioned in Fig. 1(a)–(b) are based on the two-layer HV routing model, where each layer can only provide routing segments in one direction. To efficiently utilize routing resources in two PTL routing layers, Yan proposed a two-layer planar Manhattan routing model[9] that enables routing in any direction on both the top and bottom layers. With this routing model, the resultant routing width can also be decreased to 8, as shown in Fig. 1(c). While SPL reallocation and the two-layer planar Manhattan routing model have both been shown to have positive effects on routing efficiency, prior work has not investigated the potential benefits of combining these two features. Therefore, as discussed in the following section, we propose an SPL-aware multi-terminal hierarchical routing algorithm that takes advantage of both features. As shown in Fig. 1(d), our method can further reduce the resultant routing width to 5, and all the PTL connections have met their specified extension lengths.

3 JROUTER

Fig. 2 shows the flow of JRouter, which consists of three stages: (i) track-assignment-based initial routing, (ii) SPL-tree-based hierarchical routing for length-matching, and (iii) PTL region extension for unsatisfied connections. In the first stage, the primary objective is to construct an initial routing path without extension lengths. Tracks are first assigned based on the left-edge algorithm [3] and subsequently re-assigned on the top and bottom layers. Since the initial routing does not consider length-matching constraints, some connections require additional routing resources to extend their lengths. Therefore, an SPL-tree-based hierarchical routing is developed to insert the detours for PTL connections and ensure length-matching constraints based on the initial routing results. Extension lengths of multi-terminal connections are first re-arranged among different levels of the corresponding SPL tree to maximize sharing of PTL segments. Then, a maximum-flow-based model for detours insertion is implemented for each level to satisfy the re-arranged length-matching constraints. If unsatisfied connections still exist, the PTL region extension algorithm will be performed until all the connections match their specified extension lengths.

3.1 Track-assignment-based Initial Routing

During initial routing, the extension length for any connection is not taken into account. Similar to the construction of a vertical constraint graph in CMOS channel routing, a horizontal constraint graph (HCG) can be constructed in PTL region routing. We assume HCG is a directed acyclic graph. As illustrated in Fig. 3(b), HCG depicts the relation of the routing nets in Fig. 3(c), where each node corresponds to a net, and an edge from net s_2 to net s_4 indicates

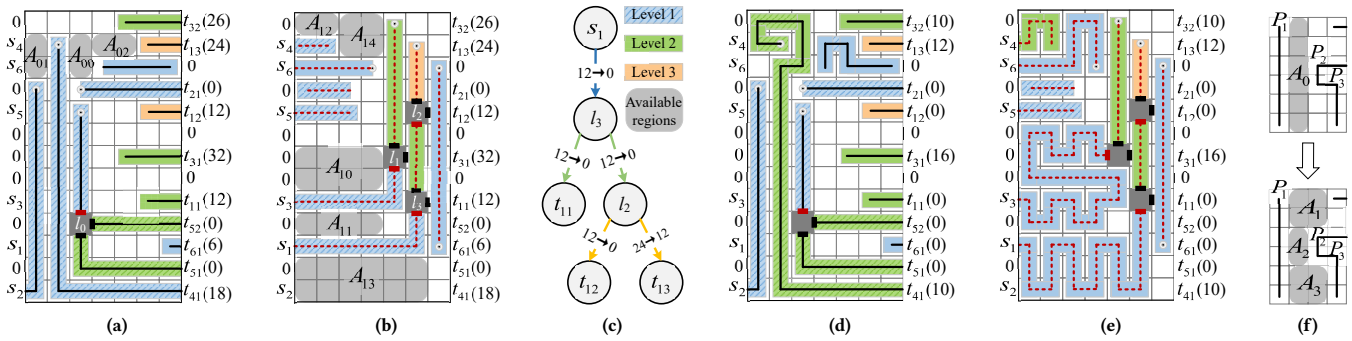

Figure 4: PTL segments with different levels and available regions on (a) top layer and (b) bottom layer. (c) SPL tree and extension lengths re-arrangement of $s_1 \rightarrow \{t_{11}, t_{12}, t_{13}\}$. Result of maximum-flow-based detours insertion for level 1 on the (d) top and (e) bottom layer. (f) A more effective method to search available regions.

the track of net s_2 must be placed to the left of the track of net s_4. Otherwise, there will be horizontal conflicts.

To minimize the initial routing width while avoiding horizontal conflicts, the left-edge algorithm [3] can be used to sort the vertical segments of all the given connections and assign them onto a minimal set of tracks as shown in Fig. 3(c). The tracks whose corresponding HCG vertex has no parents and could not cause vertical conflicts will be allocated from bottom to top. Once a net is assigned to a track, its corresponding vertex will be deleted from HCG. Then, under the two-layer planar Manhattan routing model, the tracks are re-assigned on either the top or the bottom layer. In [9], the tracks corresponding to the odd (even) column can be assigned from left to right (from right to left) on the bottom (top) layer. However, re-assignment in this way will disrupt the order of routing tracks, causing horizontal conflicts, as shown in Fig. 3(a). Therefore, a new re-assignment policy is presented, where the first half of the tracks are assigned on the top layer and the other half on the bottom layer. Fig. 3(d)–(e) show the results of our track-assignment-based initial routing on the top and bottom layers, respectively. Finally, we connect the vertical segments with their corresponding source and destination pins.

3.2 SPL-Tree-based Hierarchical Routing for Length-matching

The importance of SPL reallocation in reducing routing width was discussed in Section 2.2. Following initial routing, we propose the SPL-tree-based hierarchical routing that allocates SPLs at the branch points of connections and inserts detours to satisfy length-matching constraints. Note that an SPL tree of multi-terminals connections generally has several levels which refer to the height level from the tree's root node. To further utilize the routing resources, we re-arrange the extension lengths of multi-terminal connections among different levels of corresponding SPL trees. PTL segments with lower levels are given higher priority to insert detours, as they can yield greater savings compared to those with higher levels.

Fig. 4(c) shows the SPL tree of 4-pin connections $s_1 \rightarrow \{t_{11}, t_{12}, t_{13}\}$. The initial extension length of each edge is defined as the minimum extension length of the next-level edges connecting the same node, and the extension length of the edge connected with the leaf node is the same as that of the corresponding connection. When some extension length is inserted into the edge i, the extension length of the edge j reduces the same value if a directed path exists from i to j. For instance, 12 units of extension length are inserted at edge $s_1 \rightarrow l_3$ in Fig. 4(c). Consequently, the extension lengths of level 2 will be 0 and 0, and the extension lengths of level 3 will be 0 and 12. Therefore, the saved routing width will be maximized if the edges at lower levels insert as many extension lengths as possible. Fig. 4(a)–(b) show PTL segments with three different levels on the top and bottom layers, respectively. Levels 1, 2, and 3 are represented by blue, green, and orange segments, while gray blocks represent the available regions—blank regions to insert detours for a single PTL segment, whose height (width) must be even for vertical (horizontal) segments, respectively.

Subsequently, the hierarchical maximum-flow-based detours insertion model is used for each level of PTL segments to satisfy length-matching constraints. For horizontal PTL segments, the available regions can be calculated from top to bottom, such as A_{13} and A_{11} for PTL segment $s_1 \rightarrow l_3$ in Fig. 4(b). Similarly, it can be calculated from left to right for vertical PTL segments such as A_{01} and A_{00} for PTL segment $s_4 \rightarrow t_{41}$ in Fig. 4(a). The PTL segments that require more extension lengths have higher priority to assign available regions for reduced routing width. No available regions will be assigned for PTL segments already satisfied with length-matching, like $s_5 \rightarrow l_0$ in Fig. 4(a).

For efficient utilization of routing resources in the wiring layers, we allow the assignment of multiple available regions onto one horizontal/vertical PTL segment, instead of the restriction in [9] that a PTL segment can only use one available region for routing. As shown in Fig. 4(f), [9] only permits the assignment of A_0 to the vertical segment of P_1, thereby leaving less available regions for the vertical segments of P_2 and P_3. However, our approach enables P_1 to assign to three available regions, namely A_1, A_2, and A_3, facilitating the optimization of the maximum flow model. Besides, mutual exclusion occurs when there are two available regions in opposite directions for one PTL segment. Taking A_{01} and A_{00} as an example, if all the space of both A_{01} and A_{00} is assigned to PTL segment $s_4 \rightarrow t_{41}$, a reasonable detour can not be found in this situation. Therefore, the smaller one is set to be illegal for this iteration. If these two available regions are the same size, choose one as illegal randomly. As a result, there are two illegal available regions, A_{00} and A_{11}, in the case of Fig. 4(a)–(b).

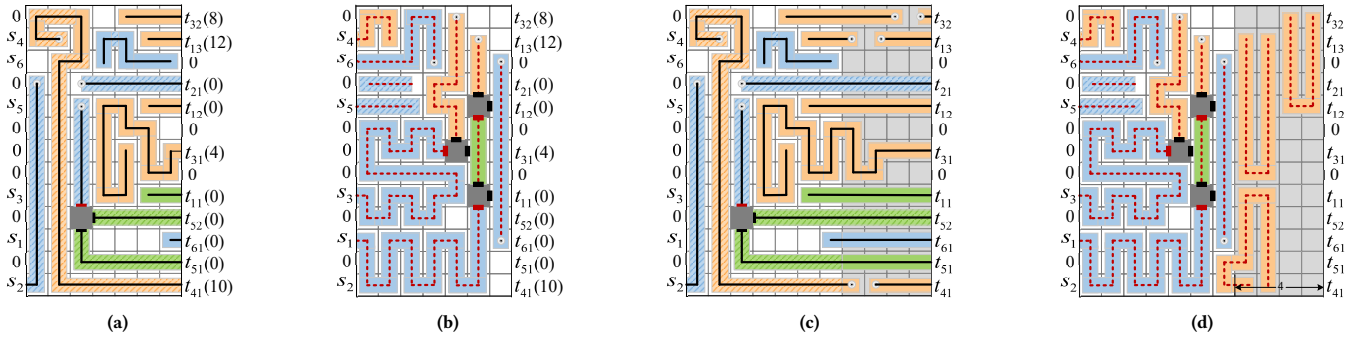

Figure 5: Results of SPL-tree-based hierarchical routing for length-matching on (a) top layer and (b) bottom layer for level 3. Results of the PTL region extension algorithm on (c) top and (d) bottom layer with the routing width increased by 4.

The process for determining the number of extension lengths to insert for each available region involves constructing a maximum-flow model. In the capacity-constrained flow graph, the vertex set represents unsatisfied connections, available regions, a super source, and a super sink. The edge set represents the candidate relation from the connection to the legally available regions. The capacity of a directed edge is set as the necessary extension length on the unsatisfied connection or the size of each available region. For a more detailed construction process of the capacity-constrained flow graph, one can refer to [9]. After one iteration of the region searching and maximum-flow-based detours insertion, another iteration will begin if there are still available regions after detours insertion, and this will continue until there are no available regions for unsatisfied PTL segments. The results of detours insertion for level 1 after three iterations are illustrated in Fig. 4(d)–(e), where 16 units of extension length are inserted to $s_3 \rightarrow l_1$, leading to a reduction of the required extension lengths of $s_3 \rightarrow t_{31}$ and $s_3 \rightarrow t_{32}$ by 16. The extension length calculations of other connections follow a similar approach. Upon completing the insertion process of one level, the level of the remaining unsatisfied segments, excluding those that end in SPL like $s_3 \rightarrow l_1$, is increased by 1 for further insertion, such as $s_4 \rightarrow t_{41}$. The process of available region search and maximum-flow-based detours insertion is then repeated until the max level is reached.

3.3 PTL Region Extension for Unsatisfied Connections

Fig. 5(a)–(b) show the results of SPL-tree-based hierarchical routing. However, four connections still have not met their required extension lengths. Thus, the PTL region needs to be enlarged to create more space for inserting detours on these unsatisfied connections. To solve this issue, we propose the PTL region extension algorithm, which consists of the following four steps.

(1) Add two columns outside the right boundary of the PTL region and assign two units of the horizontal segment to connect the corresponding pins on the top layer for the satisfied connections.
(2) Sort the unsatisfied connections in descending order according to their remaining extension lengths. Then, select the connection with the longest extension length, and insert detours on the bottom layer. If there is no space to assign detours at the bottom layer for the connection, connect its corresponding pin on the top layer.
(3) Perform available region searching and maximum-flow-based detours insertion illustrated in Section 3.2 until no space is available for the unsatisfied connections or all the connections have been satisfied with their extension lengths.
(4) If there are still unsatisfied connections repeat step (1).

Specifically, in step (2), the insertion direction is determined based on the relative positions of the first and second connections in the sorted set of unsatisfied connections. If the available upward space is greater than the downward for the first connection and the condition $row_1 + h < row_2$ is met, where h is the height of insertion space and row_1 and row_2 represent the destination rows of the first and second connection, respectively, the insertion direction will be upward. Otherwise, the insertion direction will be downward. The same applies when the available down space is greater than the upward for the first connection, except that the formula becomes $row_1 - h > row_2$. Fig. 5(c)–(d) show the final results for the top and bottom layers, respectively. All connections satisfy the length-matching constraints after two iterations of PTL region extension.

4 EXPERIMENTAL RESULTS

For a fair evaluation of the effectiveness and efficiency of our routing method, we apply the same benchmarks as other length-matching routing solutions, which contain a 16-bit RSFQ Sklansky adder and randomly generated routing instantiates of a PTL region.

4.1 Routing of a 16-bit RSFQ Sklansky Adder

The performance of JRouter is first evaluated on the 16-bit RSFQ Sklansky adder design. The adder is a 7-stage-pipelined RSFQ circuit with 7 PTL regions represented by RR_0 to RR_6, and the maximum fan-out of the circuit is 2. Table 2 shows the information of the adder routing problem, where "Height", "Connections", and "Ave/Max" refer to the routing height, the number of connections, and the average and maximum extension lengths for each PTL region, respectively. We perform JRouter on the PTL regions from RR_0 to RR_6 and compare it with two baselines, i.e., Kito's [5] and Kou's [6] routing algorithms. We did not compare JRouter with [9, 10] due to the absence of experimental results for this adder.

The experimental results are shown in Fig. 6. Compared with [5] and [6], we successfully achieved a total of 35.71% and 22.46% reduction in routing width for the entire circuit, respectively. The

Table 2: 16-bit RSFQ Sklansky adder routing information.

PTL Region	Height	Connections	Ave/Max
RR0	322	83	0 / 0
RR1	340	91	21.91 / 46
RR2	348	99	28.26 / 56
RR3	352	103	40.78 / 130
RR4	354	105	94.02 / 252
RR5	356	99	128.30 / 370
RR6	318	50	67.60 / 346

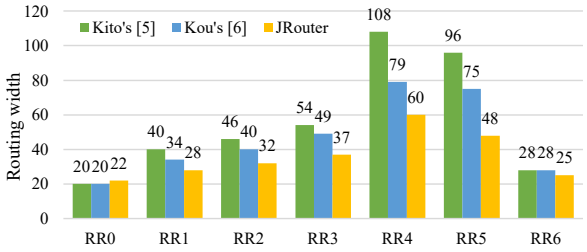

Figure 6: Routing results of the 16-bit Sklansky adder.

considerable reduction of routing width comes from RR_4 and RR_5, with the reduction of routing width being 44.44% (24.05%) and 50% (36%) compared to [5] ([6]), respectively. Overall, Jrouter outperforms [5] and [6] for most PTL regions, except for RR_0. According to the assumptions in [5], length-matching was not considered for the leftmost PTL region because it corresponds to the circuit inputs. Moreover, as all connections of RR_0 are 2-pin connections in this adder circuit, our proposed SPL-aware length-matching multi-terminal routing algorithm cannot show its superiority.

4.2 Randomly Generated Benchmarks

We also evaluate our algorithm on the randomly generated benchmarks, which are used for [5, 6, 9, 10]. These benchmarks are provided by [5]and consist of 2-pin connections with pre-allocated SPLs and two routing layers inside the PTL region. To construct multi-terminal connections, we pair adjacent 2-pin connections and randomly select one of the two source positions as a new one, converting 2-pin connections to 3-pin connections. To facilitate comparison with previous work, we assume that SPLs are placed in one extra logic-gate column for SPL pre-allocation so that the routing width of the previous work will be plus one.

Table 3 shows the results of JRouter and other routing algorithms. The data corresponding to [5] is taken from [6] since it re-ran Kito's algorithm. In Table 3, "Width" represents the routing width of one PTL region, and "Time" indicates the CPU time in seconds to complete each benchmark. It is observed that JRouter can achieve less routing width using reasonable run time. The last row of Table 3 shows the average routing width of each algorithm normalized to JRouter. On average, our proposed algorithm achieves 38.77%, 38.20%, 21.65% and 7.01% reductions in routing width compared to Kito's [5], Kou's [6] and Yan's [9, 10] routing algorithms, respectively.

5 CONCLUSION

This paper proposed JRouter, a multi-terminal hierarchical routing algorithm to produce an efficient length-matching routing solution for RSFQ circuits inspired by SPL reallocation and the planar

Table 3: Experimental results on the benchmarks [5].

Connections /Height	Ave /Max	Kito's[5] Width /Time[1]	Kou's[6] Width /Time[1]	Yan's[9] Width /Time[2]	Yan's[10] Width /Time[2]	**JRouter** Width /Time[3]
15 / 200	43/105	15 / 0.73	15 / 0.23	12 / 0.13	11 / 0.11	9 / 0.05
15 / 200	44/122	15 / 1.10	17 / 0.10	12 / 0.14	11 / 0.12	9 / 0.05
15 / 30	8/26	17 / 2.76	17 / 0.03	14 / 0.11	11 / 0.10	10 / 0.02
15 / 30	18/79	23 / 8.47	23 / 0.07	16 / 0.13	13 / 0.11	11 / 0.02
25 / 50	12/46	23/ 4.17	24 / 0.14	20 / 0.46	17 / 0.39	17 / 0.03
25 / 50	16/51	25 / 108	24 / 0.12	20 / 0.51	18 / 0.41	18 / 0.03
40 / 80	16/49	29 / 348	29 / 0.35	24 / 2.17	20 / 1.95	18 / 0.02
40 / 80	19/56	31 / 1346	33 / 0.43	24 / 2.24	21 / 2.02	18 / 0.02
40 / 200	40/168	29 / 1345	30 / 0.59	23 / 0.79	19 / 0.63	19 / 0.05
40 / 200	42/182	31 / 1346	30 / 0.60	23 / 0.84	19 / 0.68	19 / 0.09
50 / 100	23/99	43 / 1346	40 / 2.11	33 / 4.25	27 / 2.73	27 / 0.10
50 / 100	25/111	44 / 1346	40 / 2.09	33 / 4.39	27 / 2.82	24 / 0.11
Average Width (%)		163%	162%	128%	108%	**100%**

[1] Runtime on Intel(R) Xeon(R) 2.60 GHz CPU machine with 128 GB memory.
[2] Runtime on Intel Core i7-3770 3.40 GHz CPU machine with 32 GB memory.
[3] Runtime on Intel Core i7-11700 2.50 GHz CPU machine with 16 GB memory.

Manhattan routing model. Based on the HCG, tracks are initialized and re-assigned to the two-layer PTL region. An SPL-tree-based hierarchical routing algorithm is then implemented to insert detours for PTL connections. If any unsatisfied connections remain, the PTL region extension algorithm is employed. To evaluate the effectiveness and efficiency of JRouter, it is compared to state-of-the-art routing algorithms using the same benchmarks. Although the JRouter algorithm has shown superior efficiency and resource utilization compared to prior work, its performance on complex circuits has not been extensively studied. Therefore, it is crucial to explore alternative benchmarks, create new test cases, or analyze the algorithm's performance on real-world SFQ circuits to further evaluate its effectiveness.

ACKNOWLEDGMENTS

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA18000000, the National Natural Science Foundation of China under Grants 61872335, the Youth Innovation Promotion Association CAS, and the Intelligent Design Automation Lab funded by The Hong Kong Jockey Club Charities Trust.

REFERENCES

[1] W Chen, AV Rylyakov, et al. 1999. Rapid single flux quantum T-flip flop operating up to 770 GHz. *TASC* 9, 2 (1999), 3212–3215.
[2] Pei-Yi Cheng et al. 2018. Multi-Terminal Routing with Length-Matching for Rapid Single Flux Quantum Circuits. In *ICCAD*. 1–6.
[3] Akihiro Hashimoto and James Stevens. 1971. Wire routing by optimizing channel assignment within large apertures. In *Proceedings of the 8th Design Automation Workshop*. 155–169.
[4] D Scott Holmes, Andrew L Ripple, and Marc A Manheimer. 2013. Energy-Efficient Superconducting Computing—Power Budgets and Requirements. *TASC* 23, 3 (2013), 1701610–1701610.
[5] Nobutaka Kito et al. 2016. Automatic Wire-Routing of SFQ Digital Circuits Considering Wire-Length Matching. *TASC* 26, 3 (2016), 1–5.
[6] Mingyang Kou et al. 2020. Splitter-Aware Multi-Terminal Routing with Length Matching Constraint for RSFQ Circuits. *TCAD* 40, 11 (2020), 2251–2264.
[7] Ikki Nagaoka, Masamitsu Tanaka, et al. 2019. 29.3 A 48GHz 5.6mW Gate-Level-Pipelined Multiplier Using Single-Flux Quantum Logic. In *ISSCC*. IEEE, 460–462.
[8] Theodore Van Duzer and Charles William Turner. 1981. Principles of Superconductive Devices and Circuits. (1981).
[9] Jin-Tai Yan. 2020. Length-Matching-Constrained Region Routing in Rapid Single-Flux-Quantum Circuits. *TCAD* 40, 5 (2020), 945–956.
[10] Jin-Tai Yan. 2021. Via-Minimization-Oriented Region Routing Under Length-Matching Constraints in Rapid Single-Flux-Quantum Circuits. *TVLSI* 29, 6 (2021), 1257–1270.