SAGERoute 2.0: Hierarchical Analog and Mixed Signal Routing Considering Versatile Routing Scenarios

Haoyi Zhang[1], Xiaohan Gao[2,1], Zilong Shen[1], Jiahao Song[1],
Xiaoxu Cheng[3], Xiyuan Tang[4,1], Yibo Lin[1,5,6*], Runsheng Wang[1,5,6], Ru Huang[1,5,6]
[1]School of Integrated Circuits [2]School of Computer Science [3]Primarius Technology
[4]Institute for Artificial Intelligence, Peking University [5]Beijing Advanced Innovation Center for Integrated Circuits
[6]Institute of Electronic Design Automation, Peking University, Wuxi, China

***Abstract*—Recent advances in analog and mixed-signal (AMS) circuit applications call for a shorter design cycle and time-to-market period. Routing is one of the most time-consuming and tedious steps in the AMS design cycle. A modern AMS routing should simultaneously consider versatile routing scenarios (e.g., analog routing, digital routing, inter-analog-digital routing) to shoot for outstanding performance. Most previous studies only focus on one of the routing scenarios and ignore the synergism among different routing scenarios, lacking holistic and systematic investigation. In this work, we propose a hierarchical routing engine to handle the complex routing requirements in AMS circuits. By leveraging the carefully designed routing kernels hierarchically, the framework can generate high-quality routing solutions for real-world AMS circuits.**

***Index Terms*—analog routing, digital routing, inter analog-digital routing, hierarchical routing, versatile scenarios**

I. Introduction

AMS circuits play a very important role in our daily life that call for a shorter design cycle. As IC design technology advances, layout design gradually becomes the bottleneck of the AMS design flow. Routing is one of the most tedious and intricate steps in the layout design stage. Figure 1 indicates that a well-designed AMS routing must be capable of dealing with versatile routing scenarios including analog, digital, and inter-analog-digital routing. Analog routing often has sufficient routing resources but needs to be subject to complicated constraints from geometry and performance requirements [1], [2], [3], [4]. Digital routing [5], [6], [7] is usually more regular compared with analog routing, mainly targeting to improve routability and wire length under limited routing resources. Inter-analog-digital routing should connect the flexible analog part and the regular digital part while guaranteeing signal consistency. As routing requirements vary from scenario to scenario, meeting versatile requirements simultaneously in intricate AMS circuits is very challenging.

The literature has explored techniques to tackle complicated AMS routing. Most previous works focus on analog routing, such as LAYGEN [8], [9], SAGERoute [10], GeniusRoute [11]. These studies are mainly dedicated to improving layout performance in analog routing. It is not easy to directly migrate these methodologies from analog to AMS circuits while keeping the high routing quality. Although ALIGN [12], [13] and MAGICAL [14], [15] are capable of dealing with AMS circuits directly, the underlying routing algorithm is still based on pure analog routing. They do not meet the special requirements of digital parts and inter-analog-digital parts. Thus, these works are not yet enough to accommodate versatile scenarios in real design.

Given the hardness of general AMS routing, many studies target routing problems for specific circuits and applications. OpenSAR [16] proposes a special SAR-ADC architecture with template-based components and hence generates routing solutions in a specified way. OpenRAM [17] tries to automate memory layout generation for regular memory cells. AuxcellGen [18] automates the generation of analog and memory unit cells. However, these AMS routing algorithms are limited to specific circuits and applications, which is not enough to meet the demands of versatile AMS applications.

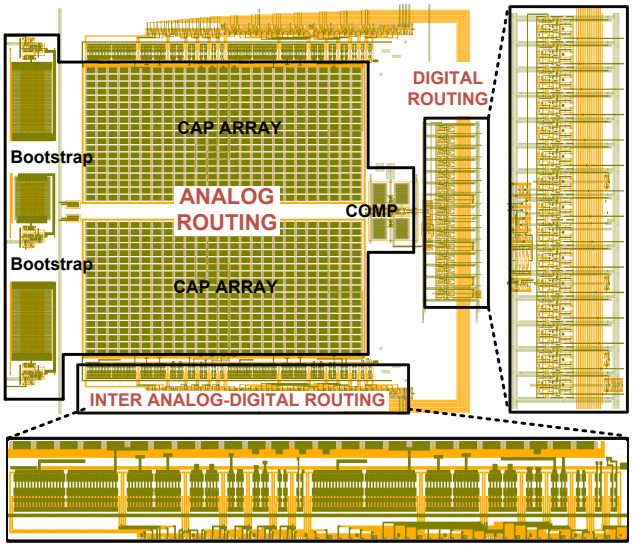

Fig. 1: A real-world AMS circuit layout with versatile routing scenarios considered.

In this paper, we propose a hierarchical routing framework considering versatile routing scenarios for general AMS integrated circuits. To reduce the difficulty of AMS routing, our framework breaks down the complicated AMS routing problem into three sub-problems including analog, digital, and inter-analog-digital routing. For each sub-problem, a dedicated routing kernel is designed to meet the particular routing features. Contributions of our framework are summarized as follows.

- We propose a hierarchical routing framework based on a novel cross-subcircuit net splitting algorithm, leveraging three different routing kernels, including analog, digital, and analog-to-digital routing kernels.
- We develop an inter-analog-digital routing algorithm that guarantees signal consistency between different signal control nets in the same group.
- We extend the analog routing with a novel accesspoint selection method targeting better routing flexibility.
- Experimental results show that our framework can generate high-quality routing solutions for complicated AMS circuits with com-

*Corresponding author: yibolin@pku.edu.cn

979-8-3503-4859-0/DATE24/© 2024 EDAA

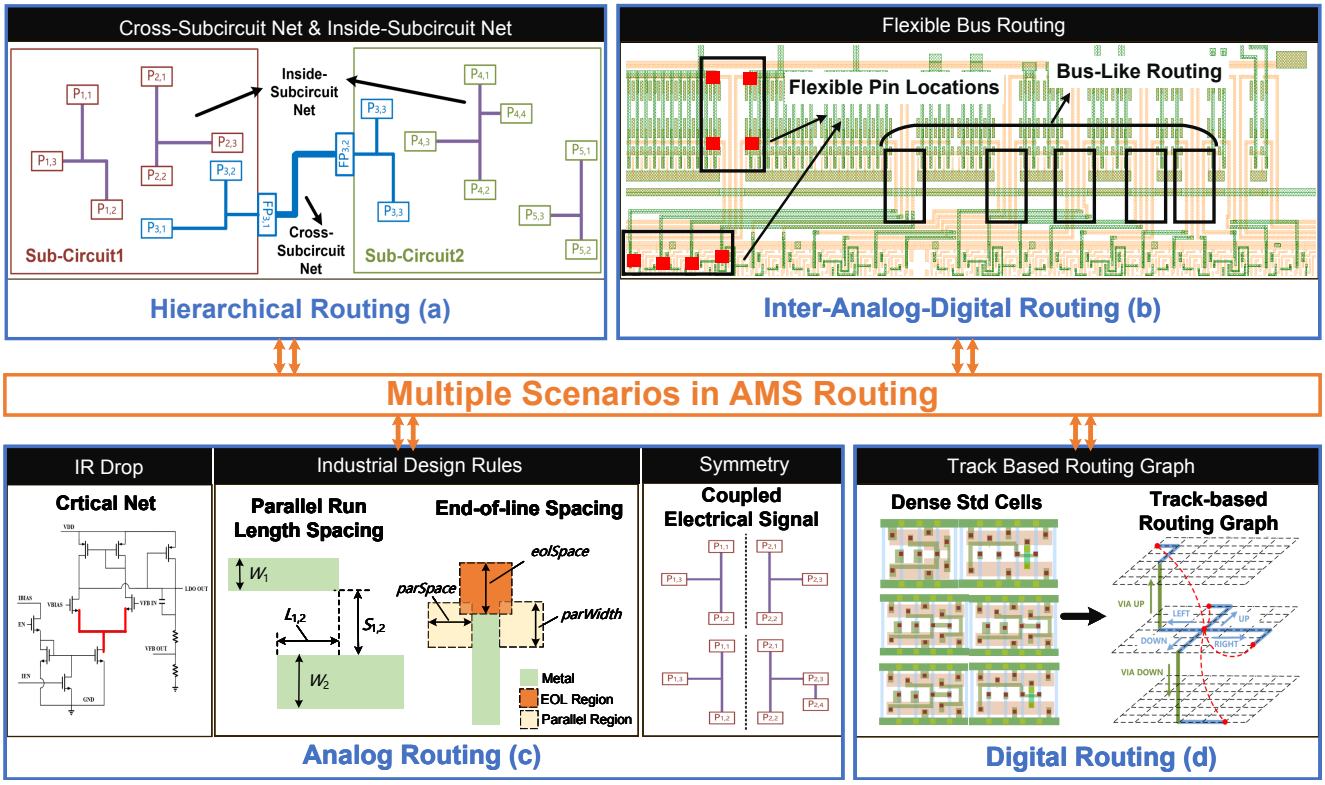

Fig. 2: Different routing scenarios in AMS circuits.

petitive post-layout performance, and demonstrate the feasibility of our framework in taped-out AMS designs.

The remaining sections are organized as follows. Section II outlines the preliminaries and formulates the AMS routing problem. Section III details our AMS routing algorithm. Section IV shows the experimental results, and Section V makes a brief conclusion.

II. Preliminaries

In this section, we will first introduce the basic background of hierarchical routing, inter-analog-digital routing, analog routing, and digital routing, respectively. Then we will formulate the complicated AMS routing problem.

A. Hierarchical Routing

Hierarchical routing aims to generate routing solutions for system-level design in an appropriate hierarchy, which is consistent with the way human designers draw layouts. It can generate better routing solutions than directly routing on a flattened design like many previous works [10], [11]. Meanwhile, there are usually cross-subcircuit nets in the system-level design as Figure 2(a) shows. These nets are usually power or signal nets which have very different wire widths in different subcircuits. In practical scenarios, AMS layout designers often set ports for cross-subcircuit nets at the boundary of subcircuits and put together the different subcircuits according to these ports, just like building blocks. In this way, the cross-subcircuit nets are split into various segments which can be easily routed with various wire widths. Automatic hierarchical routing should also honor this method.

B. Inter-Analog-Digital Routing

Inter-analog-digital routing tries to improve the routability and layout performance when dealing with the nets connecting digital and analog parts. Figure 2(b) shows a part of manually designed routing at the interface between the analog circuit and digital circuit. The analog part and digital part are likely to have a very different distribution of pin locations. These flexible pin locations lead to various topologies at the end of each net. But if we exclude the ends of these nets, it can be seen that the rest of the nets will have a very similar topology, partially like a group of bus. Such an approach not only improves routability but also guarantees signal consistency among a group of signal control nets. Although the literature has already explored conventional digital bus routing [19], [20], [21] for decades, it is hard to directly implement conventional bus routing methodology on the inter-analog-digital routing problem. The underlying reason is that conventional bus routing usually has uniform pin locations and pursues the ultimate topology match, while inter-analog-digital routing does not. Thus, inter-analog-digital routing needs a novel routing methodology to generate high-quality routing solutions.

C. Analog Routing & Digital Routing

Analog routing and digital routing are more common in physical design. Analog routing usually has sufficient routing resources but should honor versatile constraints [10], [12], [15] due to the ultimate performance requirements. Figure 2(c) shows several constraints that analog routing should consider, such as IR drop, symmetry, and industrial design rules. Digital routing has fewer constraints but inadequate routing resources. A typical solution for digital routing in a compact space is adopting routing tracks [7], [6] as Figure 2(d) shows. Constructing a track-based routing graph helps the routing kernel avoid numbers of DRVs and improves the utilization of routing resources.

D. Problem Formulation

Problem 1. Given a placement solution PL, a set of constraints $C = \{c_i | 1 \leq i \leq |C|\}$, a circuit hierarchy H, a routing type list

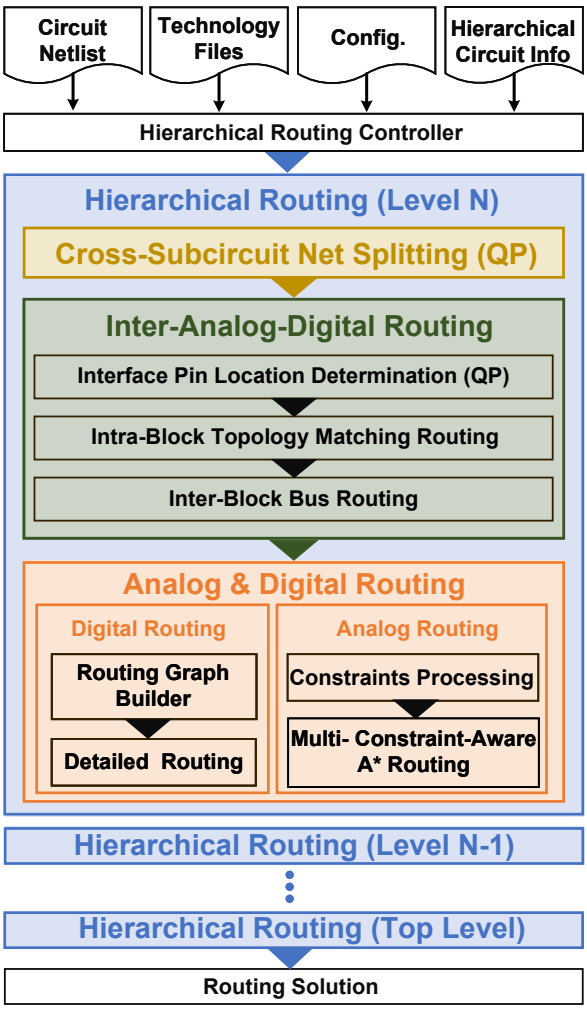

Fig. 3: Overview of hierarchical routing framework.

for subcircuits $T = \{t_i | 1 \leq i \leq |T|\}$, follow the exact routing types T to generate a routing solution $R = \{R_i | 1 \leq i \leq |R|\}$ for each subcircuit in a particular hierarchy H without any violations of constraints $c_i \in C$. (The routing type list includes different types of subcircuits like analog, digital, or inter-analog-digital.)

III. Algorithm

An overview of the hierarchical routing framework is depicted in Figure 3. The proposed framework takes the circuit netlist, technology files, user-defined configurations, and hierarchical circuit information as input. The user-defined configurations contain the necessary information for considering the constraints in multi-constraint-aware analog routing such as the IR-drop margin and symmetry net. The hierarchical circuit information includes both the basic hierarchical structure of the circuit and the routing types for each subcircuit. Detailed information on routing types is given by experienced circuit designers.

The whole routing flow is dominated by a top-level hierarchical routing controller. The main hierarchical routing takes a bottom-up strategy. In each particular hierarchy, the cross-subcircuit nets will first be split by a quadratic programming (QP) based algorithm. Then the inter-analog-digital routing kernel, analog routing kernel, and digital routing kernel will generate the routing solutions for different subcircuits separately. Inter-analog-digital routing kernel process the nets connecting analog and digital part. Analog and digital routing kernels process the normal analog and digital subcircuits.

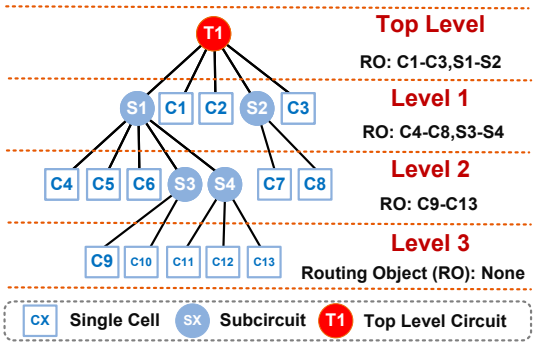

Fig. 4: Hierarchical routing tree structure.

A. Hierarchical Routing

1) Hierarchical Routing Sequence : Hierarchical routing follows a specified routing tree structure as Figure 4 shows. For each hierarchy, there are several subcircuits and cells. Subcircuits usually consist of several cells and lower-level subcircuits. But bottom subcircuits are only made up of cells. The cell represents a standard cell in a digital block or a single device in an analog block such as MOSFET, capacitor, or resistor.

The basic routing sequence follows a bottom-up strategy and only subcircuits will be routed in each hierarchy. For example, there are no subcircuits in level 3, and therefore nothing will be routed in level 3. Level 1 contains subcircuits $S1, S2$, and the routing objects will be all of the cells and subcircuits belonging to $S1, S2$ including $S3, S4$, and $C4 - C8$. $T1$ is the only "subcircuit" in the top level and the whole routing solutions for the entire AMS circuit will be generated at this level.

2) Cross-Subcircuit Net Splitting : Cross-subcircuit net splitting strategy is detailed in Algorithm 1. The basic idea of Algorithm 1 is splitting the net according to the subcircuits (line 4). For each subcircuit except the top-level subcircuit, a fake pin on the boundary will be selected as the interface to other subcircuits. Obviously, the fake pin location will greatly influence the quality of the final routing solutions. We formulate the fake pin selection problem to a QP problem [22], [23] (line 5). The fake pin should be located on the boundary of the subcircuit and we solve an independent QP problem for each edge of the boundary. Equation 1 shows the situation that the fake pin is on the bottom boundary, where x_i, y_i denotes the real pin coordinates, the x, y denotes fake pin coordinates and b^{xl}, b^{xh}, b^{yl} denotes the subcircuit boundary coordinates. After solving QP problems for each edge, we will select a relatively optimal but DRC-clean location for the fake pin from the candidates (line 6). Finally, the pins in the particular subcircuit will be replaced by the fake pin during the following fake pin selection procedure (lines 8-10).

$$\begin{aligned} &\min \sum_{\forall i} (x_i - x)^2 + (y_i - y)^2 \\ &x \geq b^{xl}, x \leq b^{xh}, y = b^{yl} \quad (bottom) \end{aligned} \tag{1}$$

Figure 5 shows an example of the cross-subcircuit net splitting algorithm. There are three subcircuits in the appropriate routing sequence. In step one, the algorithm selects the fake pin for subcircuit 1 considering all pins in the net. In step two, when selecting the next fake pin for subcircuit 2, the processed pins in subcircuit 1 (grey pins in the figure) will be represented by the fake pin of subcircuit 1. If

Algorithm 1 Cross-Subcircuit Net Splitting Algorithm

Require: Cross-subcircuit net n, Subcircuits in correct routing order SC, Subcircuits boundaries B.
Ensure: Fake pins for each subcircuit FP.

```
1: function CROSS-SUBCIRCUIT NET SPLITTING (n)
2:     P ← GET_PINS_IN_NET (n)
3:     SC_top ← GET_TOP_SUBCIRCUIT (SC, n)
4:     for SC_i ∈ SC ∧ SC_i ≠ SC_top do
5:         FP_i ← QP_SOLVER (P,B_i)
6:         DRC_CLEAN (FP_i)
7:         FP ← FP + FP_i
8:         P_sc ← FIND_PINS_IN_SUBCIRCUIT (SC_i, n)
9:         P ← P - P_sc
10:        P ← P + FP_i
11:    end for
12: end function
```

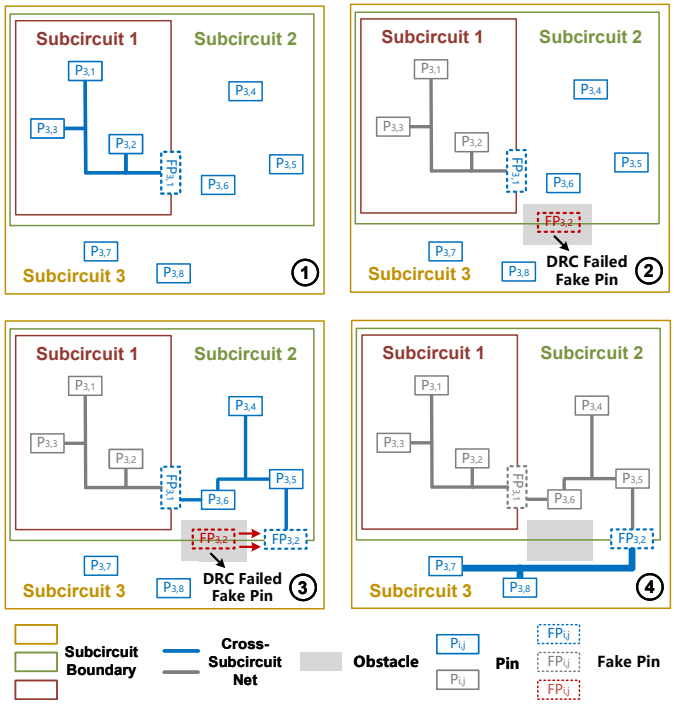

Fig. 5: An example of the cross-subcircuit net splitting.

the solutions of the QP solver lead to a DRC failure, the algorithm will greedily find a DRC-clean solution with minimum movement as shown in step three. Finally, the higher-level subcircuits are likely to have a larger wire width.

B. Inter-Analog-Digital Routing

Algorithm 2 details the proposed inter-analog-digital routing. First, the pins from the group of nets N will be divided into two groups analog pins (P_{ana}) and digital pins (P_{dig}) (line 2). Then, we also formulate the fake pin selection procedure into a QP problem (lines 3-6). After getting the available locations for fake pins, an inter-block routing is performed to generate the bus-like routing solutions (line 7). In the inter-block routing, we bundle these bus nets into one net and leverage the A-star-based routing kernel to generate the routing solution. The wire width of this net is the sum of the wire widths of all bus nets. At last, the P_{ana} and fake analog pins (FP_{ana}), P_{dig} and fake digital pins (FP_{dig}) will be connected by the intra-block routing respectively (line8-9). Intra-block routing generates fake nets by matching the real pins and fake pins one by one (lines 12-13). These fake nets will also be routed by an A-star-based routing kernel (line 14).

Algorithm 2 Inter-Analog-Digital Routing Algorithm

Require: A group of inter-analog-digital nets N, Analog part boundary B_{ana}, Digital part boundary B_{dig}.
Ensure: Inter-analog-digital routing solutions R.

```
1: function INTER-ANALOG-DIGITAL_ROUTING (N)
2:     P_ana,P_dig ← PIN_DIVIDER (P ∈ N)
3:     FP_ana ← QP_SOLVER (P_ana,B_ana)
4:     DRC_CLEAN (FP_ana)
5:     FP_dig ← QP_SOLVER (P_dig,B_dig)
6:     DRC_CLEAN (FP_dig)
7:     R ← INTERBLOCK_ROUTER (FP_ana,FP_dig)
8:     R ← INTRABLOCK_ROUTER (FP_ana,N,B_ana)
9:     R ← INTRABLOCK_ROUTER (FP_dig,N,B_dig)
10: end function
11: function INTRABLOCK_ROUTER (FP,N,B)
12:     P ← GET_PINS_IN_BLOCK(N,B)
13:     FN ← GENERATE_FAKE_NETS(P,FP)
14:     ROUTE_FAKE_NET(FN)
15: end function
```

Similar to the fake pin generation in the cross-subcircuit net splitting algorithm, the fake pin selection procedure is also formulated into a QP problem and we solve an independent QP problem for each edge of the boundary. The only difference is that we bundle the fake pins into one pin while solving the QP problem. After solving the QP problem this pin will be decomposed into a sequence of fake pins according to the bus nets.

Equation 2 depicts the details of the QP problem for the bottom edge, where $x_{i,j}, y_{i,j}$ denotes the coordinates of the j real pins from the i bus net, the x, y denotes the fake pin coordinates and b^{xl}, b^{xh}, b^{yl} denotes the subcircuit boundary coordinates. The object of QP is to minimize the quadratic distance between the fake pin and the real pins. Finally, the final fake pin with DRC-clean locations will be selected.

$$\begin{aligned} &\min \sum_{\forall i} \sum_{\forall j} (x_{i,j} - x)^2 + (y_{i,j} - y)^2 \\ &x \geq b^{xl}, x \leq b^{xh}, y = b^{yl} \qquad (bottom) \end{aligned} \tag{2}$$

Figure 6 demonstrates an example of an inter-analog-digital routing algorithm. There are two analog parts and one digital part in the example. Grouped fake pins will be generated before the whole routing flow starts. If there is an obstacle, the fake pins can avoid this area during the generation procedure. The inter-analog-digital routing will connect the real pins and fake pins in a normal way with more flexibility (red, yellow, and green lines). The inter-block nets are connected in a bus-like way (orange lines).

C. Multi-Constraint-Aware Analog Routing

Analog routing is very intricate and greatly influences the final layout performance. The ability of the analog routing kernel directly determines the quality of the final AMS layout. After thorough research, we adopt the SAGERoute [10] methodology to perform analog routing. We process versatile constraints first and then use the multi-constraint-aware analog routing kernel to generate the final routing solutions.

Different from SAGERoute [10], we propose a novel accesspoint selection method, as shown in Figure 7. Our novel accesspoint

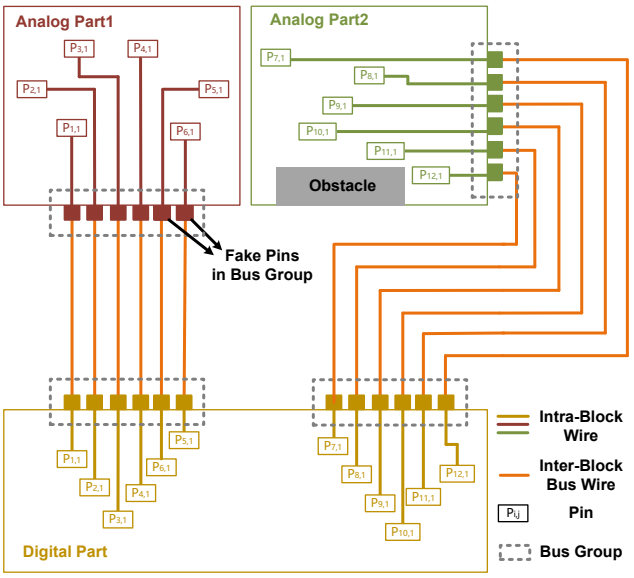

Fig. 6: An example of inter-analog-digital routing algorithm.

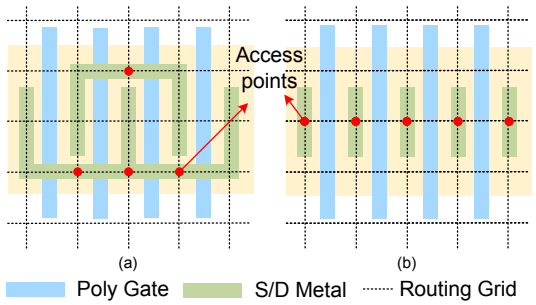

Fig. 7: Accesspoint selection methods for analog routing.

selection method can improve the routing flexibility in some scenarios. We support two kinds of accesspoint selection methods. One is a conventional method that connects all sources and drains inside the device, as Figure 7(a) shows. The router can only connect the sources or drains based on the given connections among different fingers, as most analog routers do [15], [10]. The other is a novel method that treats the sources and drains in different fingers as independent pins. This strategy allows routing to connect the sources and drains outside the particular device, which can benefit the routability in some human placements.

D. Digital Routing

Literature has already worked on digital routing for decades. The proposed digital routing follows conventional track-based routing algorithms [5], [6], [7], and the underlying algorithm is also based on the A-star [5], [7]. We extend the algorithm to support nonuniform tracks and user-defined via stack rules by constructing a nonuniform 3D routing graph. This improvement makes the digital routing more flexible and customizable which is appropriate to the digital part in AMS circuits. Due to the page limit, we omit the details.

IV. Experimental Results

The proposed hierarchical framework is implemented in C++ programming language. We adopt gurobi as the QP solver in hierarchical routing and inter-analog-digital routing. The experimental platform is a Linux server with Intel Xeon Gold 6230 CPU @ 2.10GHz. We

TABLE I: Benchmark Statistics.

Benchmark	Placement Type	Technology Node	Die Size
OTA	Automatic	TSMC40	$67.4 \times 75.7 \mu m^2$
LDO	Automatic	TSMC40	$53.3 \times 71.7 \mu m^2$
CCO	Manual	TSMC28	$56.5 \times 10.1 \mu m^2$
SAR-ADC	Manual	TSMC65	$240.6 \times 192.7 \mu m^2$

TABLE II: Comparison on Benchmarks with Automatic Placement.

Benchmark		Schematic	MAGICAL[15]	SAGERoute[10]	Our work
OTA	Gain (dB)	38.63	38.44	38.49	38.34
	UGB (MHz)	6.85	5.10	5.34	5.18
	CMRR (dB)	–	55.7	54.3	53.7
	PM (degree)	70.98	78.13	67.47	69.7
LDO	Gain (dB)	73.69	73.06	73.60	73.48
	Current (uA)	16.82	16.18	16.19	16.19
	PM (degree)	89.69	89.61	89.64	89.64
	VOD (mV)	539.6	1937.0	877.3	876.9
	VOU (mV)	540.2	1422.0	626.7	625.6

conduct the experiments on four real-world AMS designs. OTA and LDO are two primary AMS circuits, while CCO and SAR-ADC are advanced AMS circuits that have been taped out. The details of our benchmark are shown in Table I. These AMS circuits are in three different technology nodes (28nm, 40nm, 65nm). CCO and SAR-ADC are typical AMS circuits with obvious hierarchy. We support routing on both automatic placement and manually-designed placement. All the routing solutions are generated within 1 minute, which is much faster than manual efforts.

We perform a post-layout simulation with Cadence Spectre and Ultra APS to evaluate the quality of routing solutions. Calibre PEX is used to extract parasitic parameters. For OTA and LDO, we consider both resistance and capacitance (R+C+CC). For CCO and SAR-ADC, we consider the parasitic capacitance and coupling capacitance (C+CC) with the balance of simulation time and accuracy. These settings refer to the designers' experience in the real tape-out flow.

OTA and LDO are 40nm technology node. The placement results are given by the automatic placer from MAGICAL [14]. The comparison results are listed in Table II. For OTA, our work has a close performance to MAGICAL [14] and SAGERoute [10]. For LDO, our work and SAGERoute [10] have better gain and PM with slightly higher current. By leveraging wire sizing strategy, it can also be seen that our work and SAGERoute obtain 54% and 56 % reduction of voltage-over-down (VOD) and voltage-over-up (VOU), respectively. These two cases demonstrate that our analog routing kernel has similar performance to SAGERoute [10] and outperforms MAGICAL [14].

CCO and SAR-ADC are in 28nm and 65nm technology nodes respectively. CCO is the current-control oscillator and SAR-ADC is a typical system-level circuit. The placement solutions of these two cases are extracted from taped-out manual designs. MAGICAL [14] does not support 28nm and 65nm technology nodes. SAGERoute [10] ends up with routing failure when processing the inter-analog-digital part of SAR-ADC. Thus, we do not include the performance result of SAGERoute for SAR-ADC for comparison.

As shown in Table III, for CCO, our work outperforms the SAGERoute [10] with higher frequency and similar power consumption. Note that the frequency of schematic simulation is much higher in the table since no parasitic capacitance is considered. For SAR-ADC, we regard the entire circuit as the routing object, including bootstrap, comparator, CDAC (capacitor array), and SAR-control logic. Many previous works [15], [10] can only handle the analog part of such a system. To be more compatible with manual placement, we keep

TABLE III: COMPARISON ON BENCHMARKS WITH MANUAL PLACEMENT.

Benchmark		Schematic	Manual	SAGERoute[10]	Our work
CCO	Power (nW)	324.2	324.1	324.0	324.0
	Freq (MHz)	16.70	8.27	7.29	7.66
SAR-ADC	Delay (ns)	25.87	33.9	Fail	34.23
	SINAD (dB)	65.62	67.45		66.79
	ENOB (bit)	10.61	10.91		10.80
	Pcore (uW)	203.7	382.4		386.0
	FoM (fJ/conv)	5.223	7.945		8.657

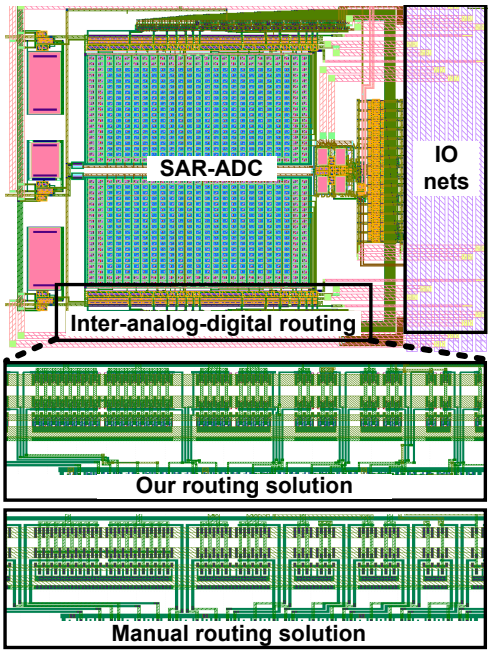

Fig. 8: Final layout of SAR-ADC.

some IO wires on the top level which will be connected to the IO Pad in a real taped-out chip. FoM represents power consumption per conversion, which is the smaller, the better. As Table III depicted, our work obtains a decent performance which is very close to the manual layout. The final layout solutions have the same area as the manual placement results.

Figure 8 shows the final layout of SAR-ADC. As figure 8 shows, the layout of SAR-ADC has a clear partition of different subcircuits. It can be seen that these subcircuits often have cross-subcircuit nets and different routing types. The proposed hierarchical routing framework can split the cross-subcircuit nets and generate different types of routing solutions by leveraging versatile routing kernels. The analog and digital interface of the SAR-control logic is zoomed in detail. It can be seen that the routing solution given by our routing framework in a few seconds has a similar routing topology to the manual layout drawn in several hours, which demonstrates the effectiveness of our inter-analog-digital routing algorithm.

V. CONCLUSION

In this paper, we propose a hierarchical analog and mixed-signal routing framework considering versatile routing scenarios. The framework can synergistically tackle analog routing, digital routing, and inter-analog-digital routing in a given design hierarchy. By handling versatile routing scenarios with dedicated routing kernels, the framework is able to generate high-quality routing solutions for AMS circuits. We validate our framework on real-world AMS designs in 28nm, 40nm, and 65nm technology nodes. Experimental results demonstrate the robustness and effectiveness of our framework.

ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation of China (Grant No. 62141404, 62125401), the Natural Science Foundation of Beijing, China (Grant No. Z230002), and the 111 project (B18001).

REFERENCES

[1] H.-C. Ou *et al.*, "Nonuniform Multilevel Analog Routing With Matching Constraints," *TCAD*, vol. 33, pp. 1942–1954, Dec. 2014.

[2] R. Martins *et al.*, "Electromigration-aware and IR-Drop avoidance routing in analog multiport terminal structures," in *DATE*, 2014, pp. 1–6.

[3] M. Torabi *et al.*, "Electromigration- and Parasitic-Aware ILP-Based Analog Router," *TVLSI*, vol. 26, pp. 1854–1867, Oct. 2018.

[4] M. M. Ozdal *et al.*, "Maze routing algorithms with exact matching constraints for analog and mixed signal designs," in *ICCAD*, Nov. 2012, pp. 130–136.

[5] G. Chen *et al.*, "Dr. CU: Detailed Routing by Sparse Grid Graph and Minimum-Area-Captured Path Search," *TCAD*, vol. 39, pp. 1902–1915, Sep. 2020.

[6] H. Li *et al.*, "Dr. CU 2.0: A Scalable Detailed Routing Framework with Correct-by-Construction Design Rule Satisfaction," in *ICCAD*, Nov. 2019, pp. 1–7.

[7] A. B. Kahng *et al.*, "TritonRoute: an initial detailed router for advanced VLSI technologies," in *ICCAD*, Nov. 2018, pp. 1–8.

[8] N. Lourenco *et al.*, "LAYGEN - Automatic Layout Generation of Analog ICs from Hierarchical Template Descriptions," in *2006 Ph.D. Research in Microelectronics and Electronics*, 2006, pp. 213–216.

[9] R. Martins *et al.*, "LAYGEN II—Automatic Layout Generation of Analog Integrated Circuits," *TCAD*, vol. 32, pp. 1641–1654, Nov. 2013.

[10] H. Zhang *et al.*, "SAGERoute: Synergistic Analog Routing Considering Geometric and Electrical Constraints with Manual Design Compatibility," *DATE*, 2023.

[11] K. Zhu *et al.*, "GeniusRoute: A New Analog Routing Paradigm Using Generative Neural Network Guidance," in *ICCAD*, Nov. 2019, pp. 1–8.

[12] T. Dhar *et al.*, "The ALIGN open-source analog layout generator: v1.0 and beyond," in *ICCAD*, Nov. 2020, pp. 1–2.

[13] K. Kunal *et al.*, "INVITED: ALIGN – Open-Source Analog Layout Automation from the Ground Up," *DAC*, p. 4, 2019.

[14] H. Chen *et al.*, "MAGICAL: An Open- Source Fully Automated Analog IC Layout System from Netlist to GDSII," *IEEE Design & Test*, vol. 38, pp. 19–26, Apr. 2021.

[15] H. Chen *et al.*, "Toward silicon-proven detailed routing for analog and mixed-signal circuits," in *ICCAD*, Nov. 2020, pp. 1–8.

[16] M. Liu *et al.*, "OpenSAR: An Open Source Automated End-to-end SAR ADC Compiler," in *ICCAD*, Nov. 2021, pp. 1–9.

[17] M. R. Guthaus *et al.*, "OpenRAM: an open-source memory compiler," in *ICCAD*, Nov. 2016, pp. 1–6.

[18] S. Kamineni *et al.*, "AuxcellGen: A Framework for Autonomous Generation of Analog and Memory Unit Cells," *DATE*, 2023.

[19] C.-H. Hsu *et al.*, "A DAG-Based Algorithm for Obstacle-Aware Topology-Matching On-Track Bus Routing," *TCAD*, vol. 40, pp. 533–546, Mar. 2021.

[20] H.-T. Zhang *et al.*, "SAT-Based On-Track Bus Routing," *TCAD*, vol. 40, pp. 735–747, Apr. 2021.

[21] Tan Yan *et al.*, "BSG-Route: A length-matching router for general topology," in *ICCAD*, Nov. 2008, pp. 499–505.

[22] P. M. Pardalos *et al.*, "Quadratic programming with one negative eigenvalue is NP-hard," *Journal of Global Optimization*, vol. 1, pp. 15–22, 1991.

[23] W. Chaovalitwongse *et al.*, "Quadratic integer programming: Complexity and equivalent forms," in *Encyclopedia of Optimization*, 2009.