IBM Systems - EDA

LEO:Line End Optimizer for Sub-7nm Technology Nodes

Diwesh Pandey, Gustavo E. Tellez, James Leland

Outline

- Background and Motivation
- Previous work
- Problem Formulation
- Algorithms and methodology
- Experimental Results
- Conclusion

Increasing routing complexity

7nm Foundry logic

Layer	Pitch (nm)	Lithography
Fin	29-30	SAQP
Gate	56-57	SADP
Contact	~56-57	LE3
1x Metal	40 1D	SADP
1x Via	~60	LE3

7nm Foundry logic 2nd Gen

Layer	Pitch (nm)	Lithography
Fin	29-30	SAQP
Gate	56-57	SADP
Contact	~56-57	EUV
1x Metal	40 1D	SADP
1x Via	~60	EUV

7nm Foundry logic 3rd Gen

Layer	Pitch (nm)	Lithography
Fin	29-30	SAQP
Gate	56-57	SADP
Contact	~56-57	EUV
1x Metal	36 2D	EUV
1x Via	~60	EUV

5nm Foundry logic

Layer	Pitch (nm)	Lithography
Fin	20	SAQP
Gate	45-50	SADP
Contact	45-50	EUV
1x Metal	26 1D	EUV
1x Via	38	EUV

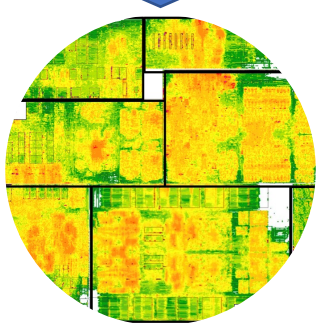
2018 → Early 2019 → Late 2019 → 2020

Source: IC Knowledge Scotten Jones

Mixed lithographic solutions used in developing sub-7nm designs

Growing chip demand forces faster time to market

- A plethora of complex design rules from multiple lithography technologies.
- Line end spacing design rules are the dominating ones across all the technologies.
- Every new class of rules requires algorithmic changes in the detail router leading to increased turn around time in chip design process.

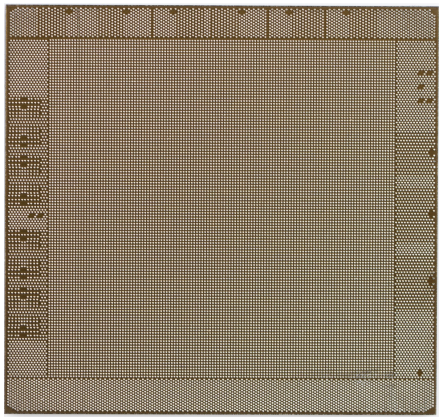


Detail routing is getting more challenging

Meeting high performance

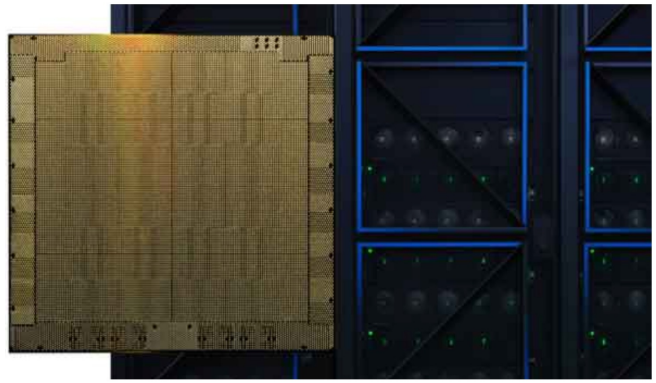
Building high performance microprocessor (IBM Telum and Power10) pushes the design density targets, thus making detail routing job even harder!

Denser designs lead to tighter line end spacing.



IBM Telum chip

- 7nm Samsung technology
- 530sqmm chip size
- 22.5 billion transistors
- 5+GHz base clock frequency



IBM Power10 chip

- 7nm Samsung technology
- 602sqmm chip size
- 18 billion transistors
- Up to 4GHz base clock frequency

https://www.ibm.com/blogs/systems/ibm-telum-processor-the-next-gen-microprocessor-for-ibm-z-and-ibm-linuxone/

Objectives

Goal:

- Develop a routing plugin independent of detail router to solve line end spacing rules.
- Simultaneously address line end optimization in two major lithography technologies: EUV and SADP

Key contributions:

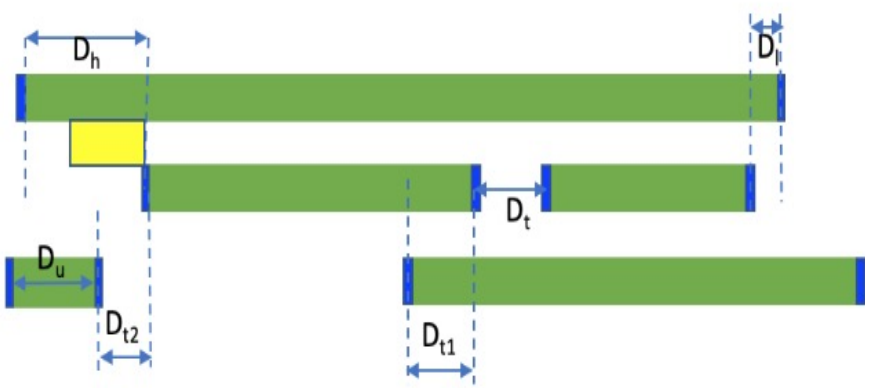
- Technology independent modelling of line ends
- constraint graph formulation and graph partitioning algorithm.
- A set of algorithms to solve constraint graph.
- Solution for routing-based line end optimization problem.
- Routing plugin-based novel design methodology flow.

Previous work

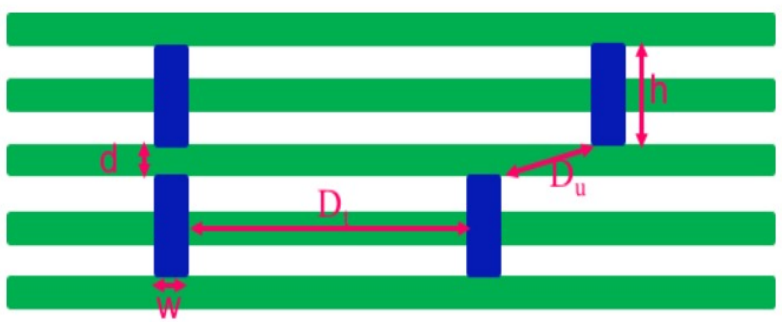
- The existing methods model line end spacing rules
  - These rules are specific to SADP or cut mask.
- These methods are integrated within detail routers.
  - This approach is costly as it disrupts the whole toolchain.
- Differences:
- None of the existing work, models EUV based line end spacing rules.
- None of the work has introduced a mechanism to model these rules for multiple lithography technologies.
- None of the work has provided a solution which is independent of the detail router, which makes is very cost effective.

Technology independent modelling

- Define a common model for EUV technology with SADP technology.
- Define artificial trims for EUV, which simulate the SADP behavior of placing trims
- The artificial trims enable one step towards mixed technology implementation
- Simulate dummy metal rule in SADP with metal gaps in EUV



EUV constraints



SADP constraints

Constraint graph

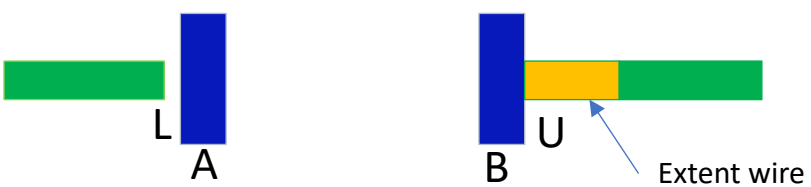
The line end optimization problem is formulated with the constraint graph:

given a graph G(V, E):

Minimize $\sum_{i \in n} W_i X_i$ **subject to: spacing constraints**

$L_i <= x_i <= U_i$.

Where x_i represents the position of the line end node. W_i represents weight of the node.

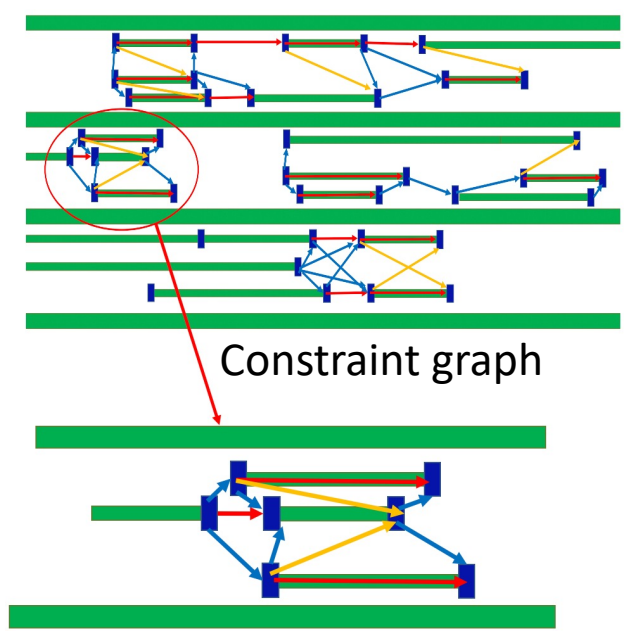


$$L \leq x_A \leq L + P$$

$$U - P \leq x_B \leq U$$

Graph partitioning

- Constraint graph can be partitioned using pruning technique.
- Feasible region of a node can be pruned around ideal location by a heuristic value: $P = k * D_t$

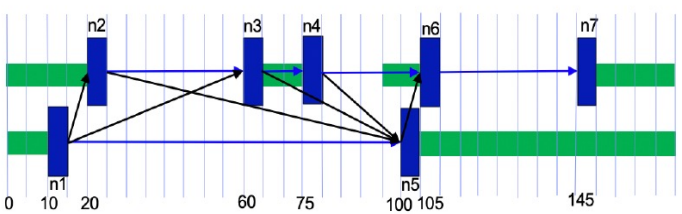


Line end optimization(LEO)

1) **Longest path-based algorithm(LPA) on subgraphs with Heuristic integer variable assignments.**
   - ❑ Highly scalable w.r.t. run time and guarantees to produce a solution if exists.

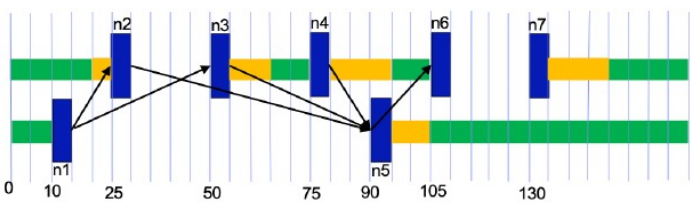
2) **Extent minimization algorithm on subgraphs**
   - ❑ Helps in pushing density limits for the designs.

1



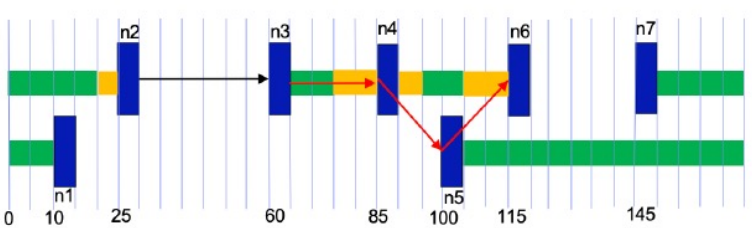
Constraint graph with initial node positions

2



Initial feasible solution after running LPA

3

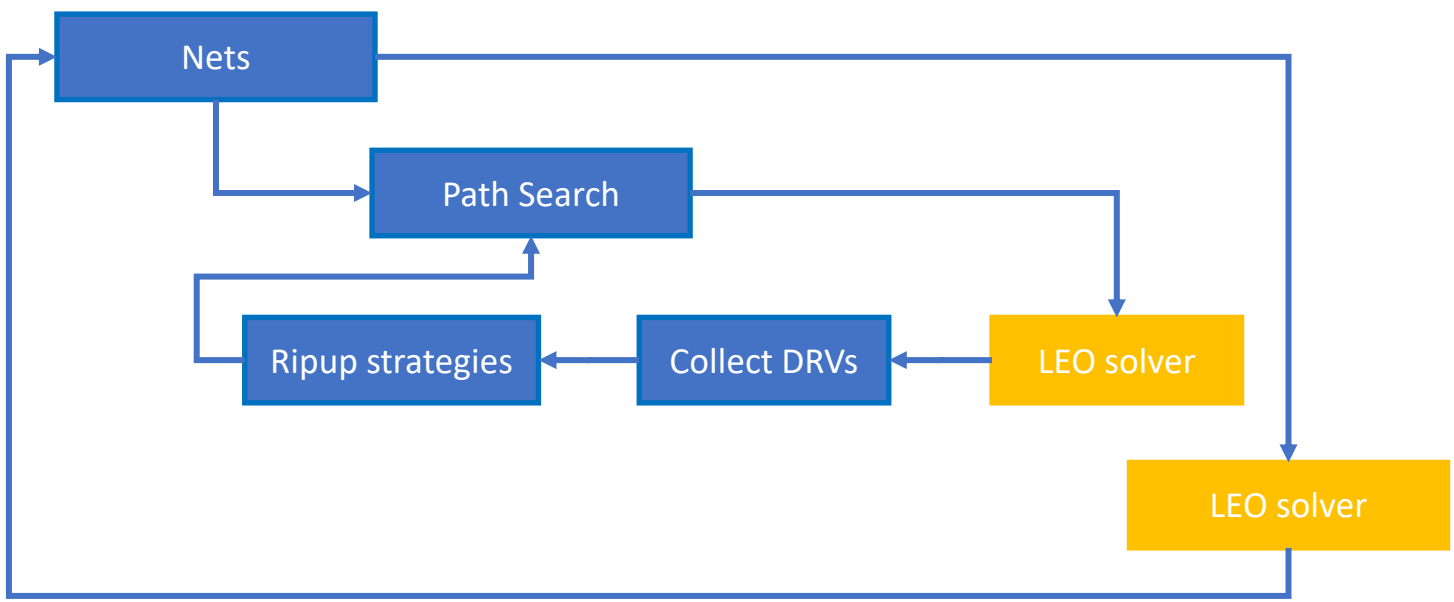


Final solution after running extent minimization algorithm

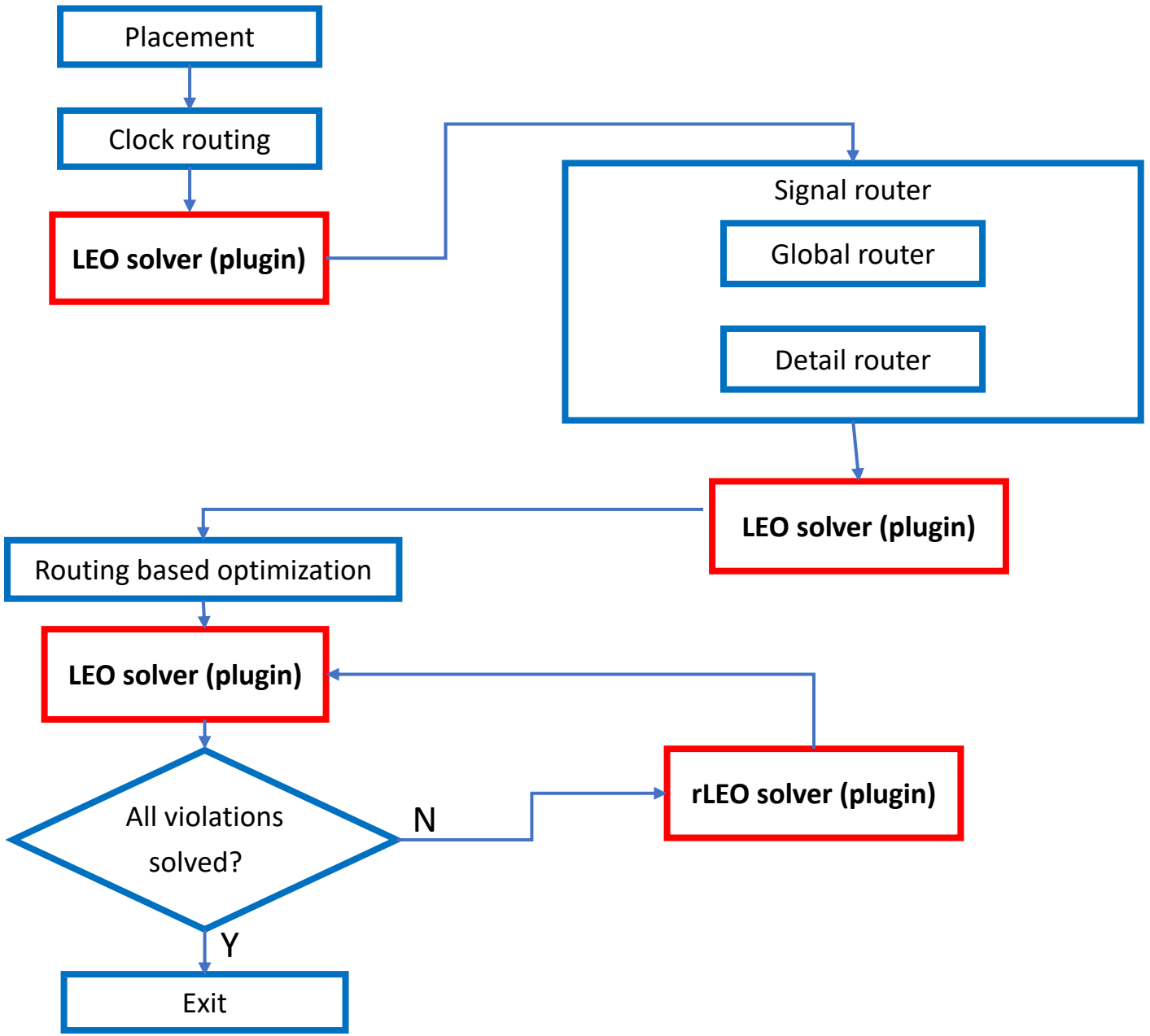
Routing based line-end optimization(rLEO)

Solve infeasible issues using ripup and reroute approach

Run LEO solver incrementally along with detail routing path search



Plugin-based methodology flow



Experimental setup

C++; Intel Xeon 2.7 GHz CPU and 126 GB memory.

Experiments run on 6 EUV and 6 SADP designs.

With 6 SADP designs we test the efficacy of LEO solver against ILP solver(CPLEX).

LEO converges much faster with a marginal increase (avg. 4%) in the wire length.

EUV benchmarks			Graph detail		
Designs	#Nets	#Gates	#Nodes	#Arcs	#Metal shapes
EUV1	234,621	210,874	6,712,374	2,918,169	3,311,686
EUV2	244,220	225,659	6,026,936	3,416,334	2,999,981
EUV3	402,688	361,437	15,413,760	5,268,430	7,697,268
EUV4	720,490	611,223	16,740,799	8,432,198	8,333,372
EUV5	861,422	793,193	19,565,996	10,906,504	9,775,062
EUV6	1,738,299	1,746,638	46,601,419	23,116,490	23,243,333

SADP benchmarks			Extent length(mm)		Extent length	Runtime(s)	
Designs	#Nets	#Gates	cplex	LEO	Inc (%)	cplex	LEO
SADP1	3680	8971	0.51	0.58	1.27	28	1.2
SADP2	9743	25948	1.39	1.45	4.31	101	4.8
SADP3	14278	45903	1.90	1.98	4.21	33	4.7
SADP4	19718	81547	2.13	2.20	3.28	50	4.5
SADP5	26754	85974	3.81	3.98	4.46	361	9.0
SADP6	30600	67573	4.97	5.20	4.62	1262	12.0

Experimental setup

The table shows the efficacy of our graph partitioning algorithm.

The largest design from the benchmark (EUV6) is used for the experiment with different prune factor(P.F.) values

P.F. value of 3 or 4 gives us the best result in terms of runtime and DRV count.

Prune Factor	Total subgraphs	Subgraphs unsolved	Subgraph unsolved %	Runtime (s)	#DRV
1	31,678,471	636,295	2.008	360	46,000
2	27,259,736	17,973	0.065	380	260
3	25,394,026	6,383	0.025	390	245
4	23,520,178	4,932	0.020	380	245
5	22,172,952	4,924	0.022	480	245
10	17,763,543	4,889	0.027	6600	245

Experimental setup

LEO solver converges very quickly and takes only 400 seconds on the largest design.

It leaves a very small set of subgraphs which are infeasible to solve.

Designs	Total subgraphs	Subgraphs unsolved	Subgraph unsolved %	Runtime (s)
EUV1	4,233,742	42	0.0010	38
EUV2	3,156,474	9	0.0003	40
EUV3	11,063,076	186	0.0017	120
EUV4	8,741,186	136	0.0016	130
EUV5	9,534,048	98	0.0010	160
EUV6	25,394,026	226	0.0009	400

Experimental setup

rLEO solver fixes most of the infeasible subgraphs with a reasonable runtime.

The overall increase in the wirelength is less than 2%.

designs	Initial WL (mm)	Final WL (mm)	Increased length%	LEO #DRV	rLEO #DRV	rLEO runtime(s)
EUV1	3,385.19	3,439.59	1.61	48	5	180
EUV2	3,666.34	3,725.84	1.62	20	0	120
EUV3	7,429.12	7,534.63	1.42	100	15	360
EUV4	10,185.17	10,358.17	1.70	180	45	700
EUV5	12,587.20	12,837.39	1.99	150	25	650
EUV6	26,829.64	27,345.69	1.92	245	60	1200

Conclusion

- We proposed a routing plugin for 7nm EUV and SADP technologies.
- Derived a common mathematical formulation for both technologies.
- A graph partitioning technique was introduced to enable parallelism
- A set of algorithms were introduced to solve the constraint graph.
- Proposed an integrated line end and detail routing solution
- Our solver always produces a legal solution if one exists.
- Solution is implemented in the development of Power10 and Telum processors.

Thank you!