EIC Lab | SKKU

NEURAL INFORMATION PROCESSING SYSTEMS

AnalogToBi: Device-Level Analog Circuit Topology Generation via Bipartite Graph and Grammar Guided Decoding

Seungmin Kim, Mingun Kim, Yuna Lee, Yulhwa Kim
Sungkyunkwan University, Republic of Korea

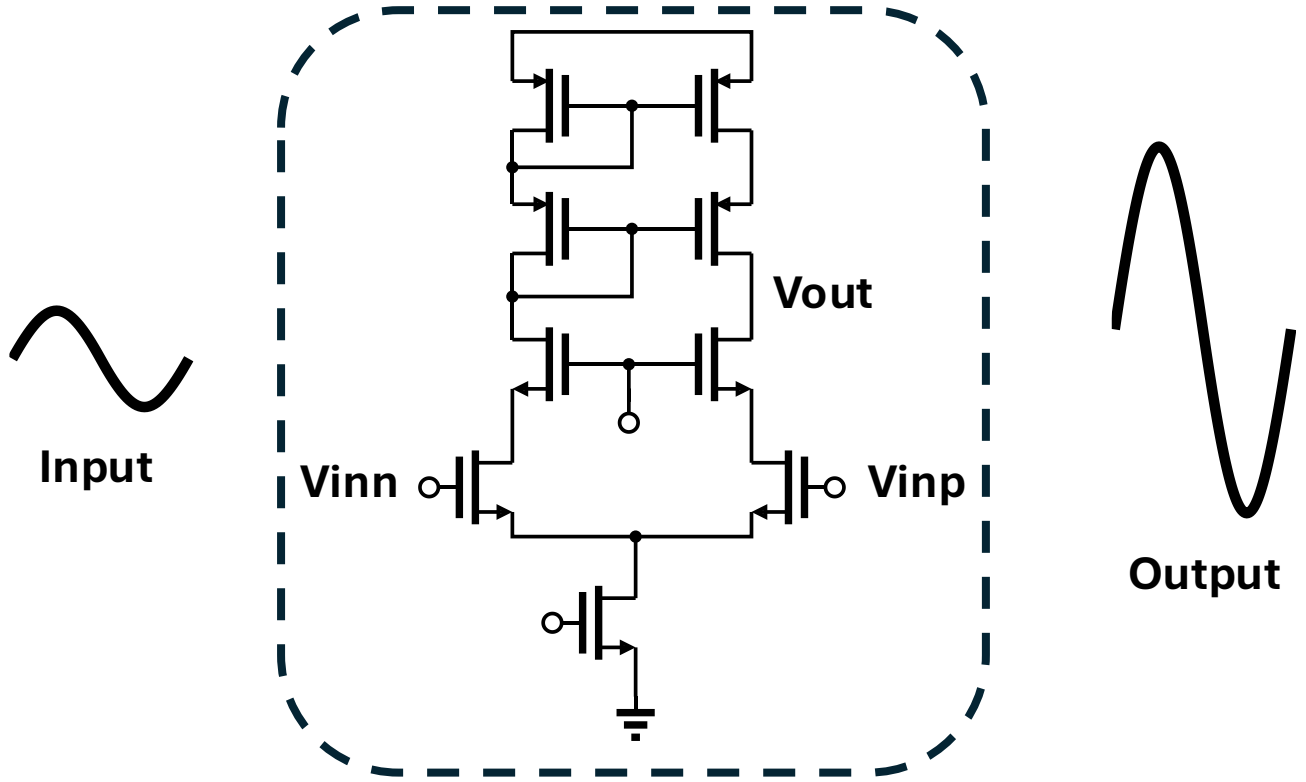
Presenter

Seungmin Kim (seungmin.kim@skku.edu)

Background

❑ Why analog circuit topology generation?

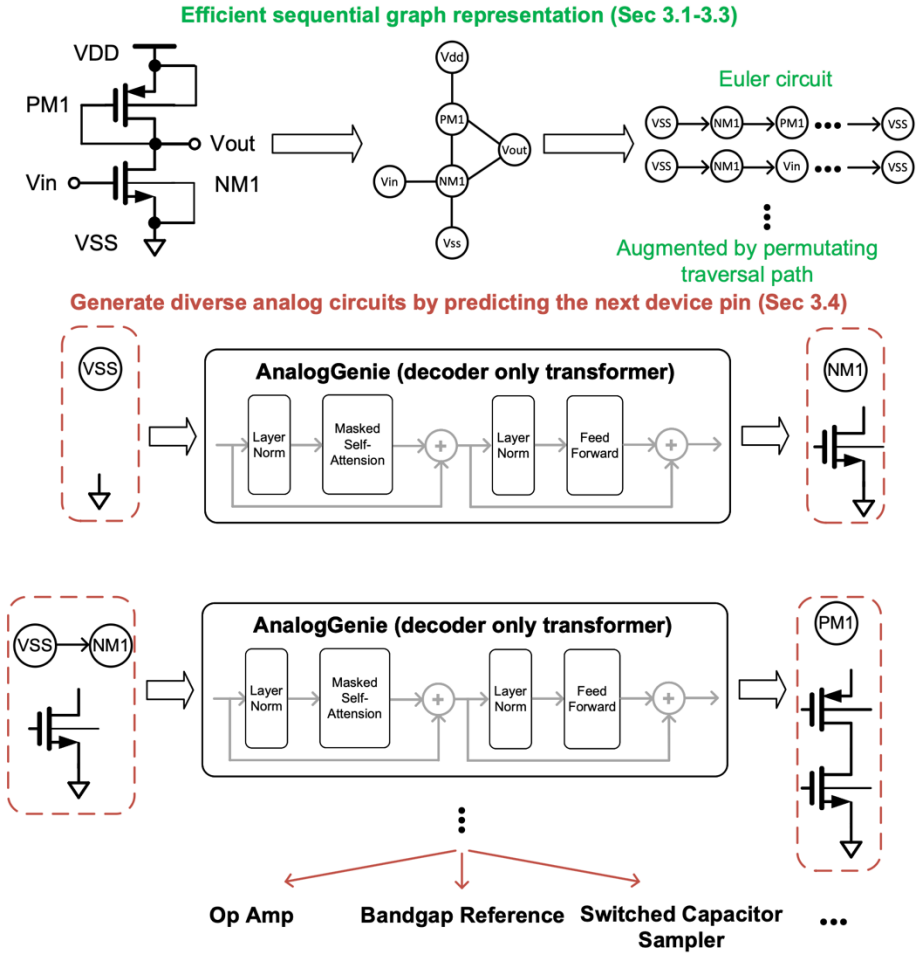
- **A circuit topology is for core functionality.**
- **For example, an OpAmp topology is designed for amplification.**



⇨ Device-level generation is suitable for discovering truly new topologies.

Background

❑ Transformer for circuit graph generation



- **A circuit is converted to graph, then to sequence.**
- **Each graph node is treated as a token.**
- **The transformer learns the graph in sequence format.**

Fig: [1] AnalogGenie, Gao et al., ICLR 25

Motivational Study

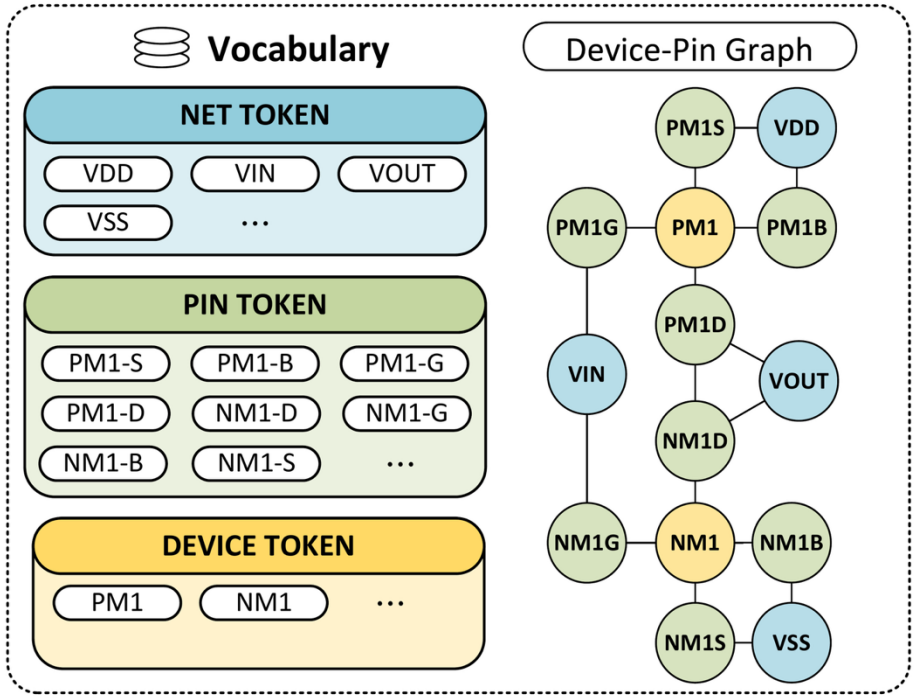
❑ Key requirement of circuit generation model

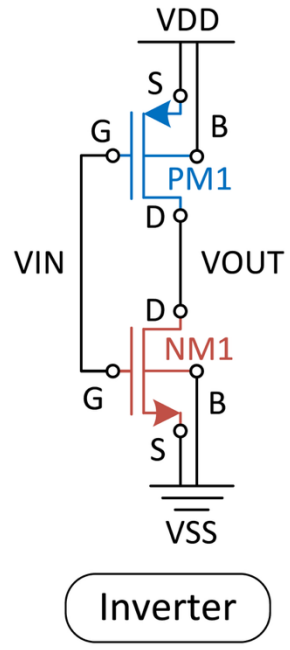
- Validity ↑: Graph constructed from sequence should be well structured, so that it can be converted to a netlist.
- Novelty ↑: Graph constructed from sequence should be non-isomorphic to dataset (new topology).
- Exact ↓: Sequences should differ from sequences in the datasets (memorization).
- Generation controllability: Generated circuit should meet the circuit type that the user intends.

Motivational Study

❑ Reproduced result of the previous work (AnalogGenie w/o HITL (Human-in-the-Loop) [1])

Validity ↑	75.6%
Novelty ↑	50.4%
Exact ↓	39.6%



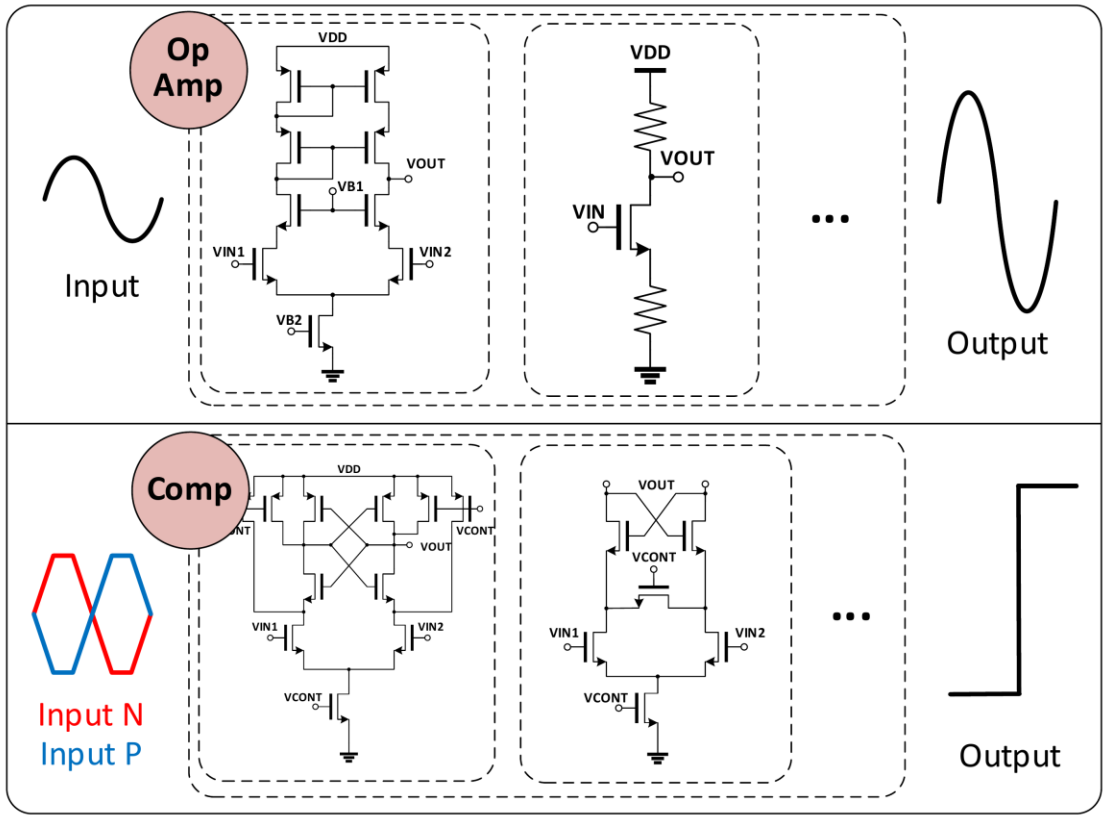


- One token mistake
  ⟹ **Invalid circuit.**
- Device-pin graph
  ⟹ **Memorization**

[1] AnalogGenie, Gao et al., ICLR 25, Source code: [URL]

Proposed Method 1.

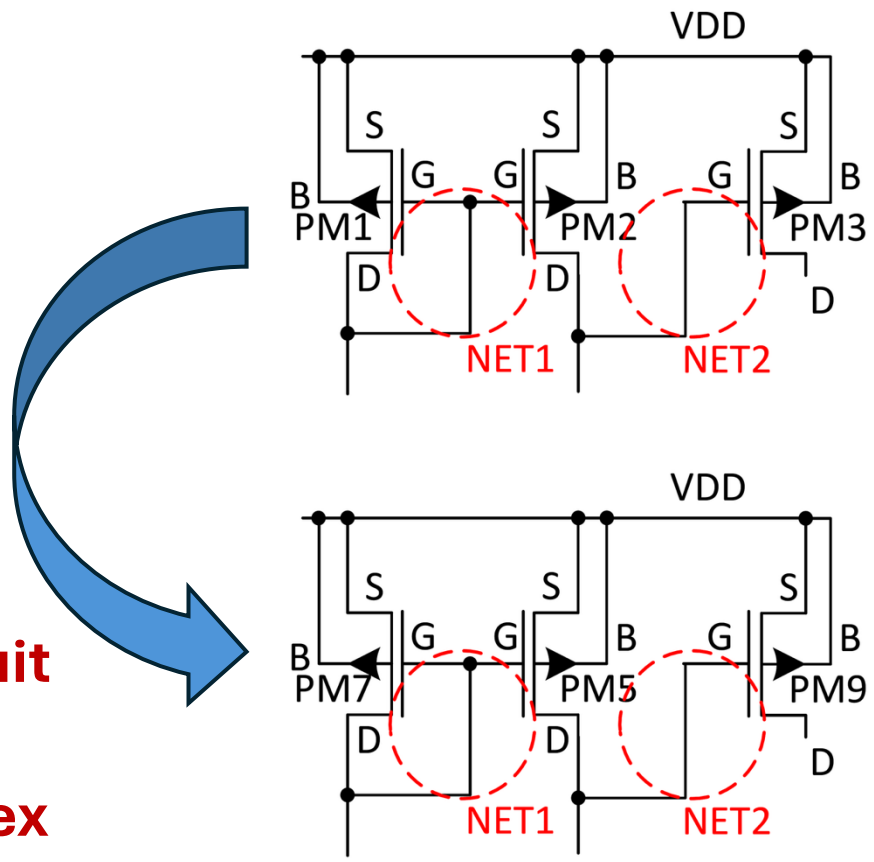
❑ Circuit Type Conditioning



- **Different functionality ⟹ Different topology.**
- **Prevent interpolation during inference.**
- **Explicit condition for the intended circuit generation.**
- **Memorization Problem**

Proposed Method 2.

❑ Device Renaming

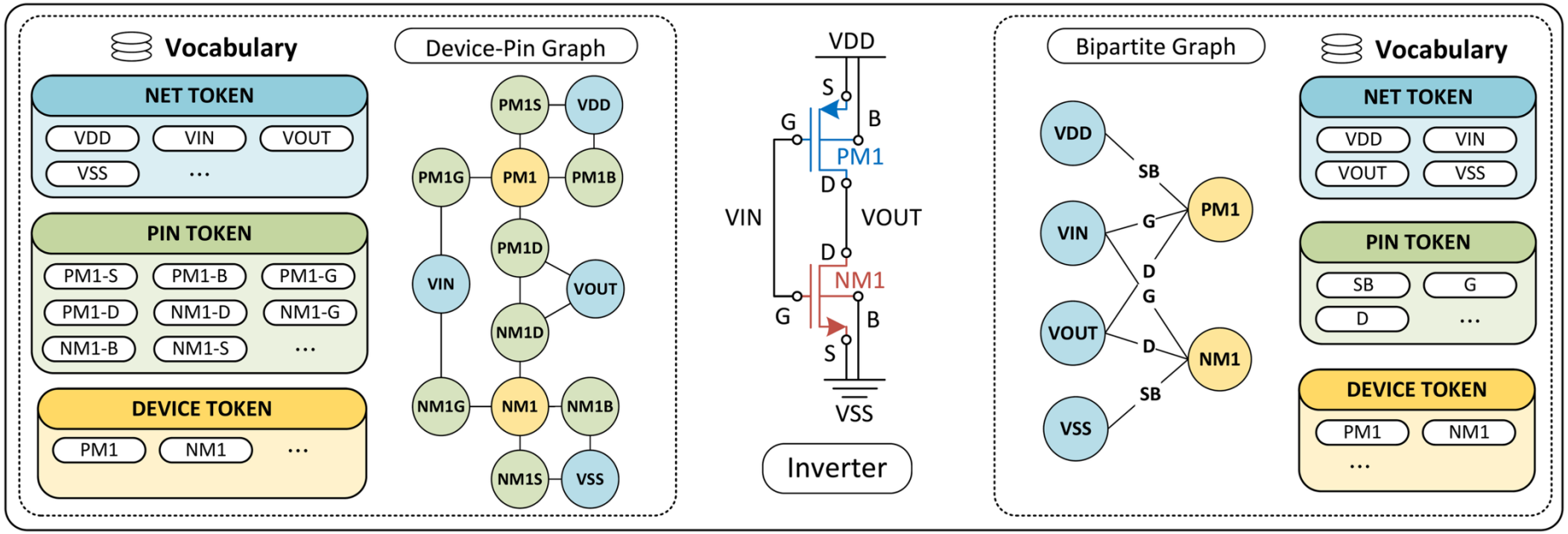


Same Circuit
But
Device Index

- **Every device with an index has a dedicated token (e.g., PM1)**
- **Permuting the indices does not change the circuit itself.**
- **Breaks the index-based pattern memorization.**
- **Validity degradation**

Proposed Method 3.

❑ Bipartite Graph Representation



Device-pin graph

4 pins per device

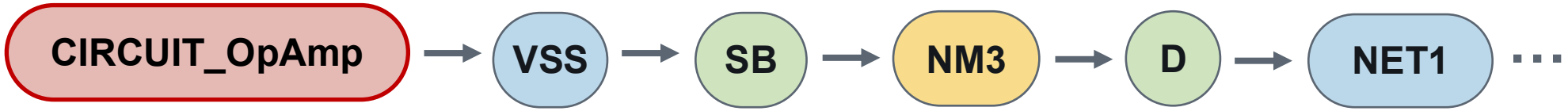
Bipartite graph

Shared pin token

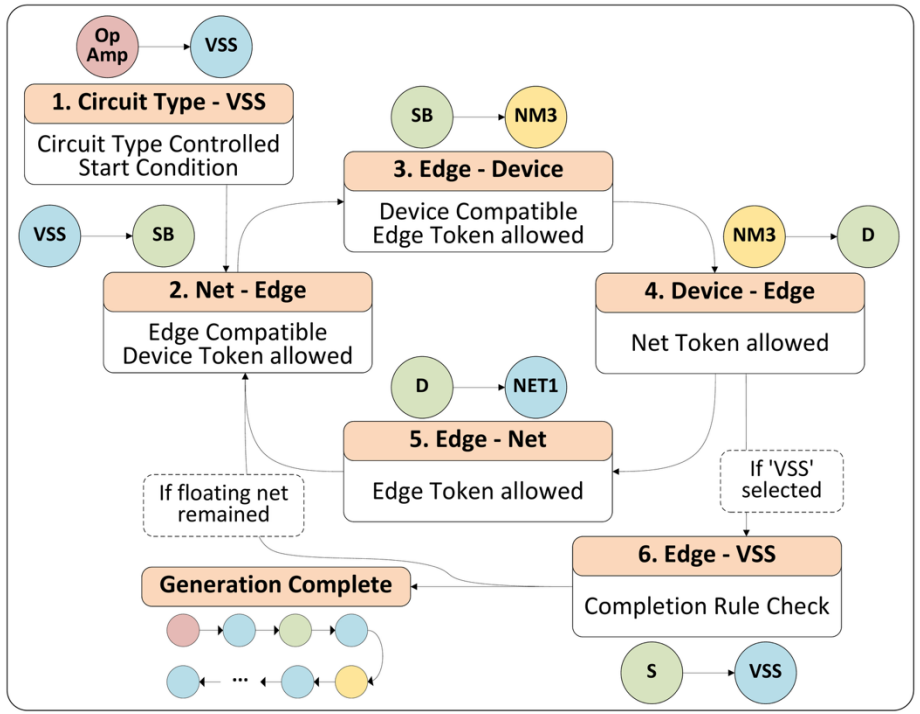
- **Bipartite Graph mitigates memorization, but is vulnerable to decoding mistakes**

Proposed Method 4.

❑ Grammar Guided Decoding



- **Sequence of bipartite graph ⟹ 'Net-Pin-Device-Pin' pattern.**



- **Treat the decoding step as a state machine transition.**
- **State: [T-1,T] tokens**
- **Prevent invalid transitions**
  ⟹ Improve validity

Evaluation

❑ Main table

Method	Properties			Evaluation Results					
	Model (#param.)	HITL	#Type	Validity↑	Novelty↑	Val.&Nov.↑	Exact.↓	Div.↑	GED↑
LaMAGIC [12]	Flan-T5 (250M)	×	1	96.0%	–	–	–	–	
AnalogGenie [13]	Custom (11.8M)	×	11	73.5%	98.9%$^{+}$	–	–	–	
AnalogGenie [13]	Custom (11.8M)	○	11	93.2%	99.0%$^{+}$	–	–	–	
AnalogGenie-Lite [28]	Custom (11.8M)	○	11	97.0%	99.8%$^{+}$	–	–	–	
AnalogGenie* [13]	Custom (11.8M)	×	11	75.6%	50.4%	26.0%	39.6%	26.0%	25.2
AnalogToBi (Proposed)	**Custom (11.3M)**	×	**15**	**97.8%**	**92.1%**	**89.9%**	**0%**	**75.3%**	**93.6**

AnalogGenie*: reproduced w/o HITL.

+ : novelty reported in the AnalogGenie paper, whose evaluation method was not described. Our novelty uses graph isomorphism.

- **Div.: Ratio of unique valid & novel topologies among generated samples.**
- **GED: Average minimum graph edit distance between each generated sample and the training set.**

Evaluation

❑ Ablation

Options				Evaluation Results			
Representation	Type Token	Rename	Grammar	Validity ↑	Novelty ↑	Val.&Nov. ↑	Exact. ↓
Device-pin	×	×	–	75.6%	50.4%	26.0%	39.6%
Device-pin	○	×	–	86.1%	29.2%	15.4%	57.5%
Device-pin	○	○	–	55.6%	84.1%	39.7%	0%
Bipartite	×	×	×	53.6%	61.0%	15.3%	0%
Bipartite	×	×	○	61.8%	57.3%	19.9%	0%
Bipartite	○	×	○	97.8%	74.6%	72.4%	1.4%
Bipartite	○	○	○	**97.8%**	**92.1%**	**89.9%**	**0%**

❑ On device-pin graph

- **Type token induces memorization (Exact. 39.6% → 57.5%).**
- **With renaming, validity degrades (Validity 86.1% → 55.6%).**

❑ Bipartite graph

- Inherently lower memorization (Exact. 0%).
- Type token & Grammar improve validity (Validity 53.6% → 97.8%).
- Orthogonally, renaming improves novelty (Novelty 74.6% → 92.1%).

Evaluation

❑ Ablation

Options				Evaluation Results			
Representation	Type Token	Rename	Grammar	Validity ↑	Novelty ↑	Val.&Nov. ↑	Exact. ↓
Device-pin	×	×	–	75.6%	50.4%	26.0%	39.6%
Device-pin	○	×	–	86.1%	29.2%	15.4%	57.5%
Device-pin	○	○	–	55.6%	84.1%	39.7%	0%
Bipartite	×	×	×	53.6%	61.0%	15.3%	0%
Bipartite	×	×	○	61.8%	57.3%	19.9%	0%
Bipartite	○	×	○	97.8%	74.6%	72.4%	1.4%
Bipartite	○	○	○	**97.8%**	**92.1%**	**89.9%**	**0%**

❑ On device-pin graph

- Type token induces memorization (Exact. 39.6% → 57.5%).
- With renaming, validity degrades (Validity 86.1% → 55.6%).

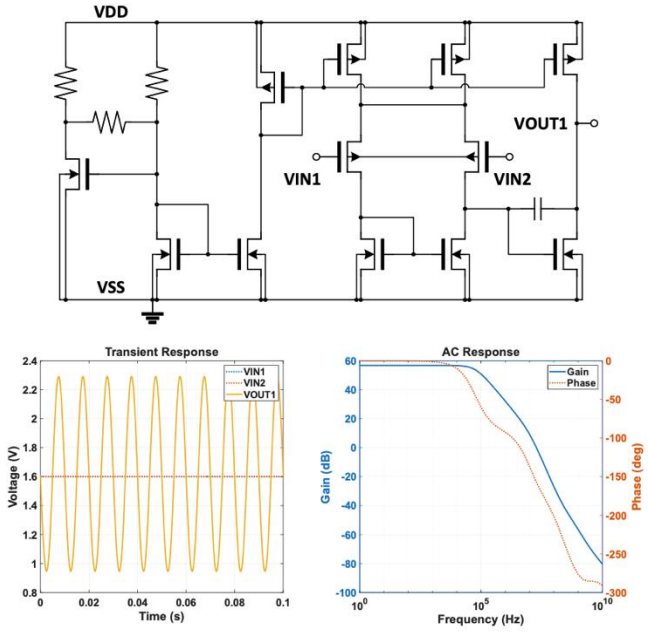
❑ Bipartite graph

- **Inherently lower memorization (Exact. 0%).**
- **Type token & Grammar improve validity (Validity 53.6% → 97.8%).**
- **Orthogonally, renaming improves novelty (Novelty 74.6% → 92.1%).**

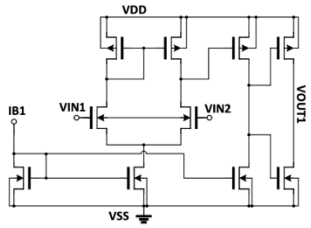
Evaluation

❑ Case Study

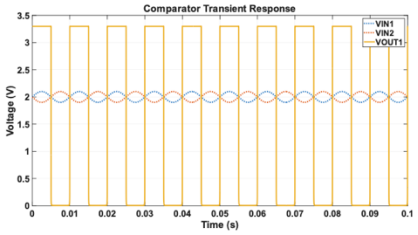
- **Type token enables systematic simulation.**



OpAmp

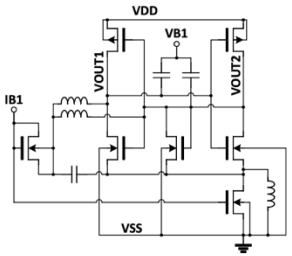


(a) Generated comparator

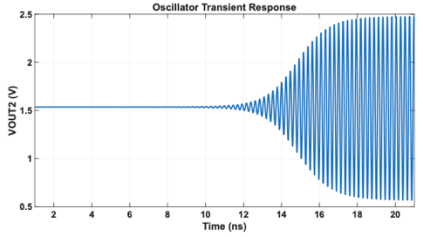


(d) Comparator transient response

Comparator

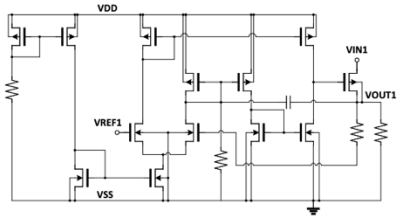


(b) Generated oscillator topology

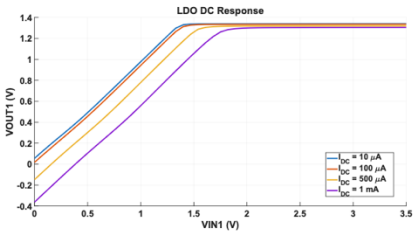


(e) Oscillator transient response

Oscillator



(c) Generated LDO topology



(f) LDO DC simulation result across current load

LDO

Thank you

Questions and comments welcome

Code & weights
github.com/Seungmin0825/AnalogToBi

Contact Info

Name
Seungmin Kim

E-Mail
seungmin.kim@skku.edu

Website
eic.skku.edu