MIXSyn: An Efficient Logic Synthesis Methodology for Mixed XOR-AND/OR Dominated Circuits

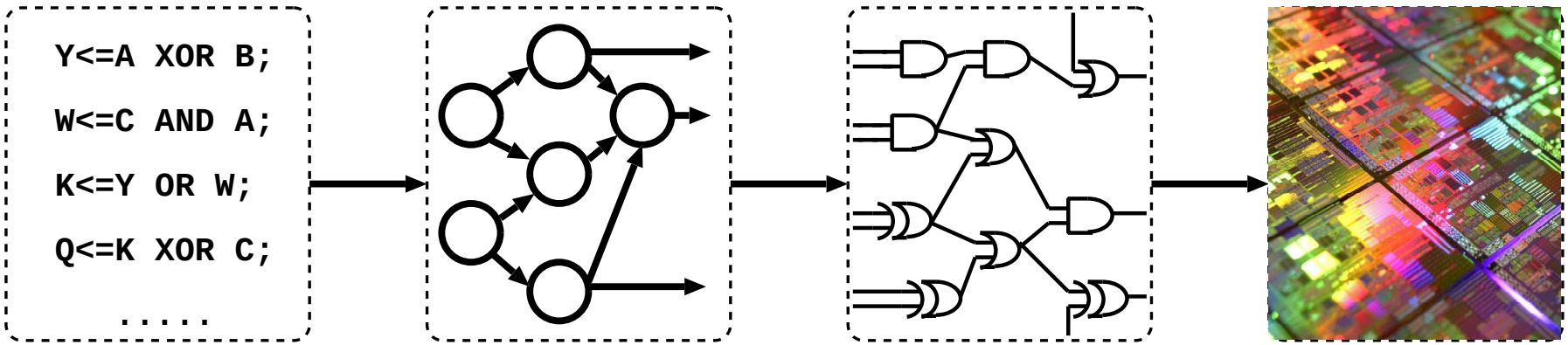
Luca Amarú, Pierre-Emmanuel Gaillardon, Giovanni De Micheli

Integrated Systems Laboratory (LSI), EPFL, Switzerland

Wednesday, January 23rd, 2013

Integrated Circuits Logic Design

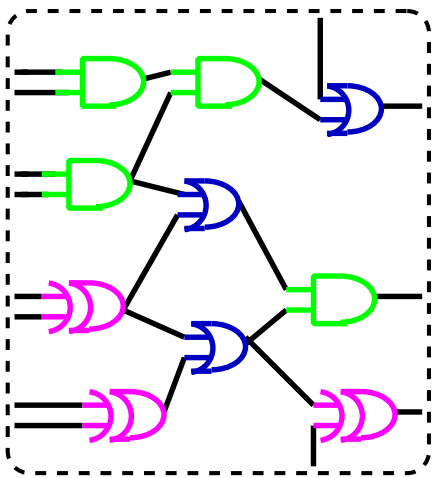
- Integrated circuits are designed using logic synthesis techniques.



- Established methods for AND/OR-dominated circuits (MIS, SIS, ..).
- Novel heuristics for XOR-dominated circuits (BDS, FLDS, ..).
- Real-life designs contain both AND/ORs and XORs intertwined.

Synthesis of XOR-AND/OR Dominated Circuits

- Open question:



How to efficiently synthesize XOR-AND/OR dominated circuits?

- We will address the question during this talk:

MIXSyn: a novel synthesis methodology

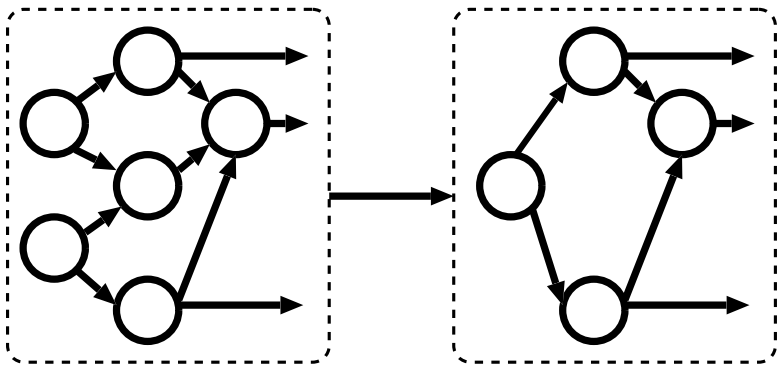
Outline

(Brief) Introduction on Logic Synthesis

Logic (*combinational*) synthesis:

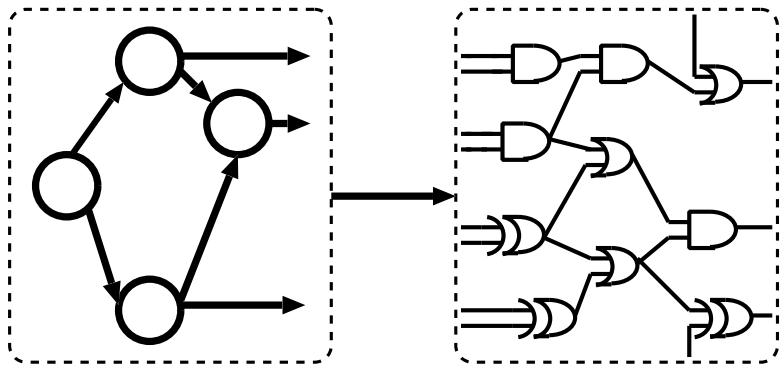
logic optimization $\Rightarrow$ **technology mapping**

- Logic optimization:



  - Compacts the original logic circuit size.
  - Original techniques exploited AND/OR representations.
  - New techniques take advantage of XOR operators.

- Technology mapping:

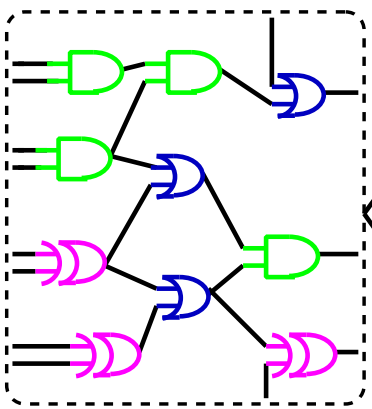


  - Transposes the logic circuit in a netlist of gates.
  - Usually targets a library of standard cells.
  - Two main operations: cell matching and selection.

Why Change Approach to Logic Synthesis?

Practical logic circuits contain AND/ORs and XORs intertwined

- Logic optimization:
  - Current heuristics efficiently manipulates
    - **[1]** AND/OR operations
    - **[2]** XOR operations
  - Which method to manipulate both AND/OR and XOR?
- Technology mapping:
  - Standard cell libraries have
    - At most 5/6 inputs.
    - 100s of gate functions.
  - However, with 5 inputs, the total number of different functions is $2^{32} >> 100s$.



XOR-Optimization?

or

AND/OR-Optimization?

Fully flexible Standard cell library

Stack limit: max 5 inputs

Logic design flexibility:

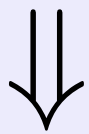
> 4 BILLIONS functions

MIXSyn Synthesis Flow 1/2

MIXSyn efficiently optimizes and maps XOR-AND/OR dominated circuits.

MIXSyn

Hybrid optimization

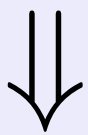


Library-free technology mapping

MIXSyn Synthesis Flow 2/2

MIXSyn

Hybrid optimization



Library-free technology mapping

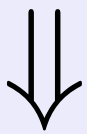
- **Hybrid optimization**
  - Efficiently manipulates both AND/OR and XOR ops.
  - Two-step procedure.
- **Library-free technology mapping**
  - Exploits all 2^{2^n} functions realizable in a n-input gate.
  - No standard cell library.
  - New XOR-AND/OR composed gates.

MIXSyn: augmented design flexibility

MIXSyn Steps

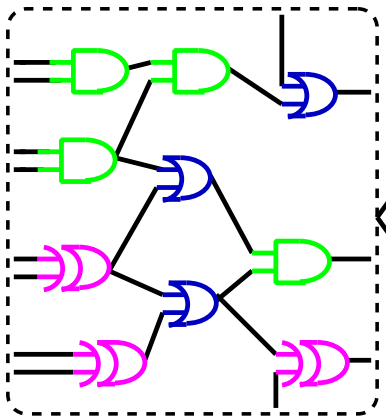
MIXSyn

Hybrid optimization



Library-free technology mapping

Hybrid Logic Optimization: Procedure



***XOR*-Optimization?**

or

***AND*/*OR*-Optimization?**

Answer:

Hybrid-Optimization

Procedure steps:

Algorithm 1 Hybrid Logic Optimization

INPUT: Input Boolean Network (IBN)

OUTPUT: Optimized Boolean Network (OBN)

XOR-extraction (phase-α).

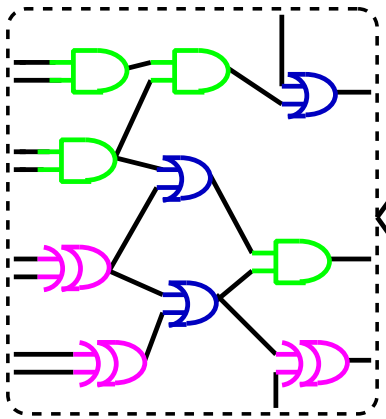
XOR-splitting (phase-β).

Controllability don't care computation (phase-χ).

AND/OR-minimization (phase-ψ).

XOR-merging (ω).

Hybrid Logic Optimization: Procedure



***XOR*-Optimization?**

or

***AND*/*OR*-Optimization?**

Answer:

Hybrid-Optimization

Procedure steps:

Algorithm 2 Hybrid Logic Optimization

INPUT: Input Boolean Network (IBN)

OUTPUT: Optimized Boolean Network (OBN)

XOR-extraction (phase-α).

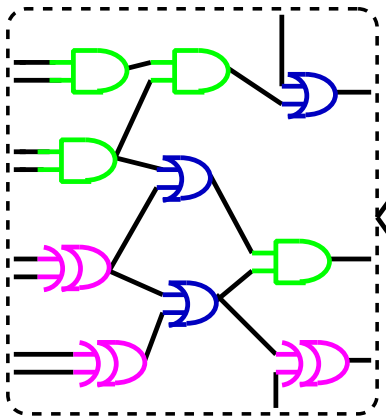
XOR-splitting (phase-β).

Controllability don't care computation (phase-χ).

AND/OR-minimization (phase-ψ).

XOR-merging (ω).

Hybrid Logic Optimization: Procedure



XOR-Optimization?

or

AND/OR-Optimization?

Answer:

Hybrid-Optimization

Procedure steps:

Algorithm 3 Hybrid Logic Optimization

INPUT: Input Boolean Network (IBN)

OUTPUT: Optimized Boolean Network (OBN)

XOR-extraction (phase-α).

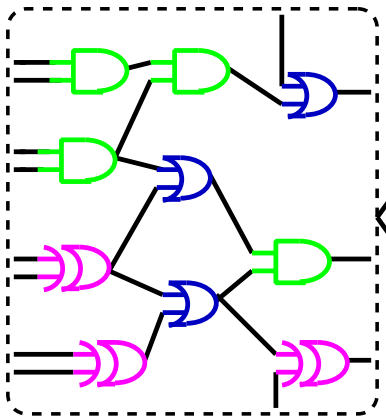
XOR-splitting (phase-β).

Controllability don't care computation (phase-χ).

AND/OR-minimization (phase-ψ).

XOR-merging (ω).

Hybrid Logic Optimization: Procedure



XOR-Optimization?

or

AND/OR-Optimization?

Answer:

Hybrid-Optimization

Procedure steps:

Algorithm 4 Hybrid Logic Optimization

INPUT: Input Boolean Network (IBN)

OUTPUT: Optimized Boolean Network (OBN)

XOR-extraction (phase-α).

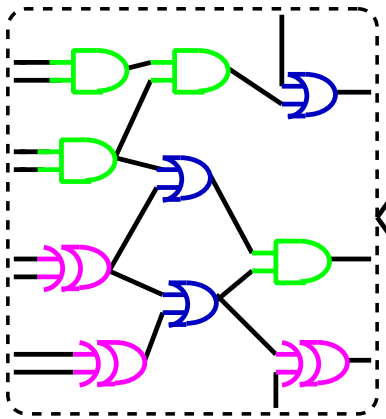
XOR-splitting (phase-β).

Controllability don't care computation (phase-χ).

AND/OR-minimization (phase-ψ).

XOR-merging (ω).

Hybrid Logic Optimization: Procedure



XOR-Optimization?

or

AND/OR-Optimization?

Answer:

Hybrid-Optimization

Procedure steps:

Algorithm 5 Hybrid Logic Optimization

INPUT: Input Boolean Network (IBN)

OUTPUT: Optimized Boolean Network (OBN)

XOR-extraction (phase-α).

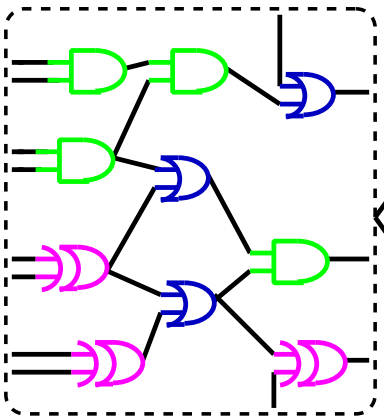
XOR-splitting (phase-β).

Controllability don't care computation (phase-χ).

AND/OR-minimization (phase-ψ).

XOR-merging (ω).

Hybrid Logic Optimization: Procedure



XOR-Optimization?

or

AND/OR-Optimization?

Answer:

Hybrid-Optimization

Procedure steps:

Algorithm 6 Hybrid Logic Optimization

INPUT: Input Boolean Network (IBN)

OUTPUT: Optimized Boolean Network (OBN)

XOR-extraction (phase-α).
XOR-splitting (phase-β).
Controllability don't care computation (phase-χ).
AND/OR-minimization (phase-ψ).
XOR-merging (ω).

Hybrid Logic Optimization: Example

Target function: $f = ab + bc + \overline{a}\overline{b} + ca$

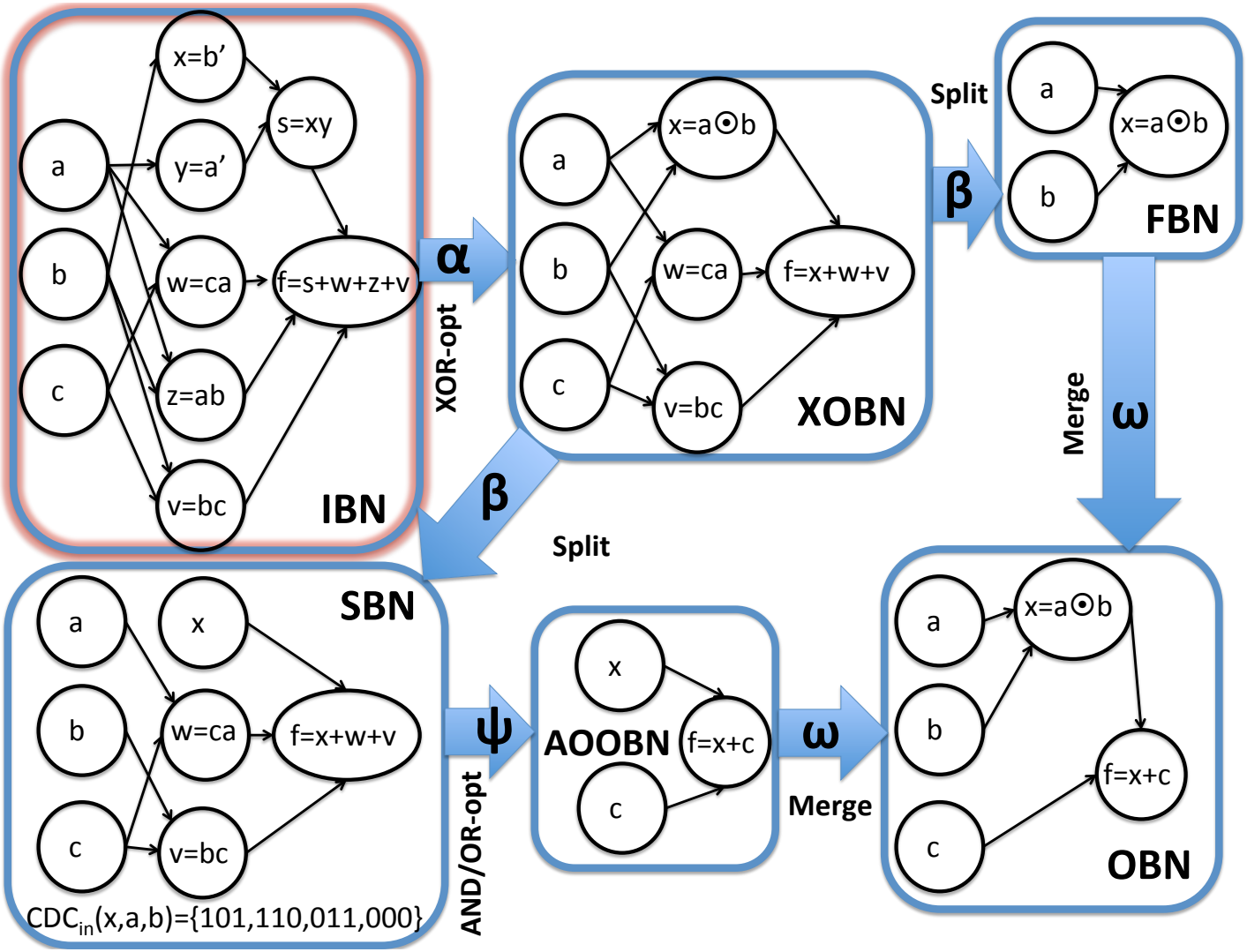
Traditional optimization techniques:

AND/OR optimization: $f = ab + \overline{a}\overline{b} + c(a + b)$

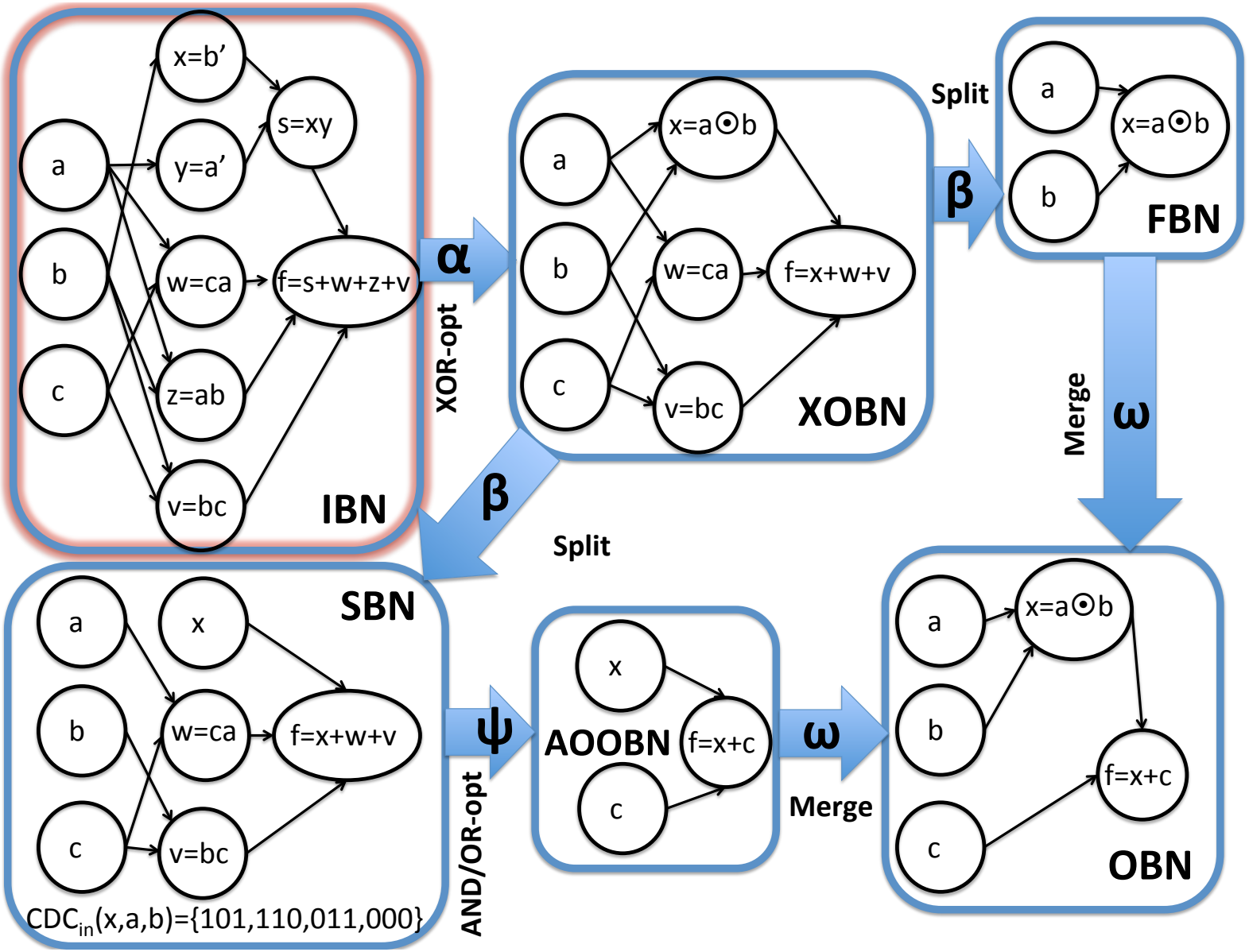
XOR optimization: $f = bc + (a \odot b) + ca$

Hybrid Logic Optimization: Example

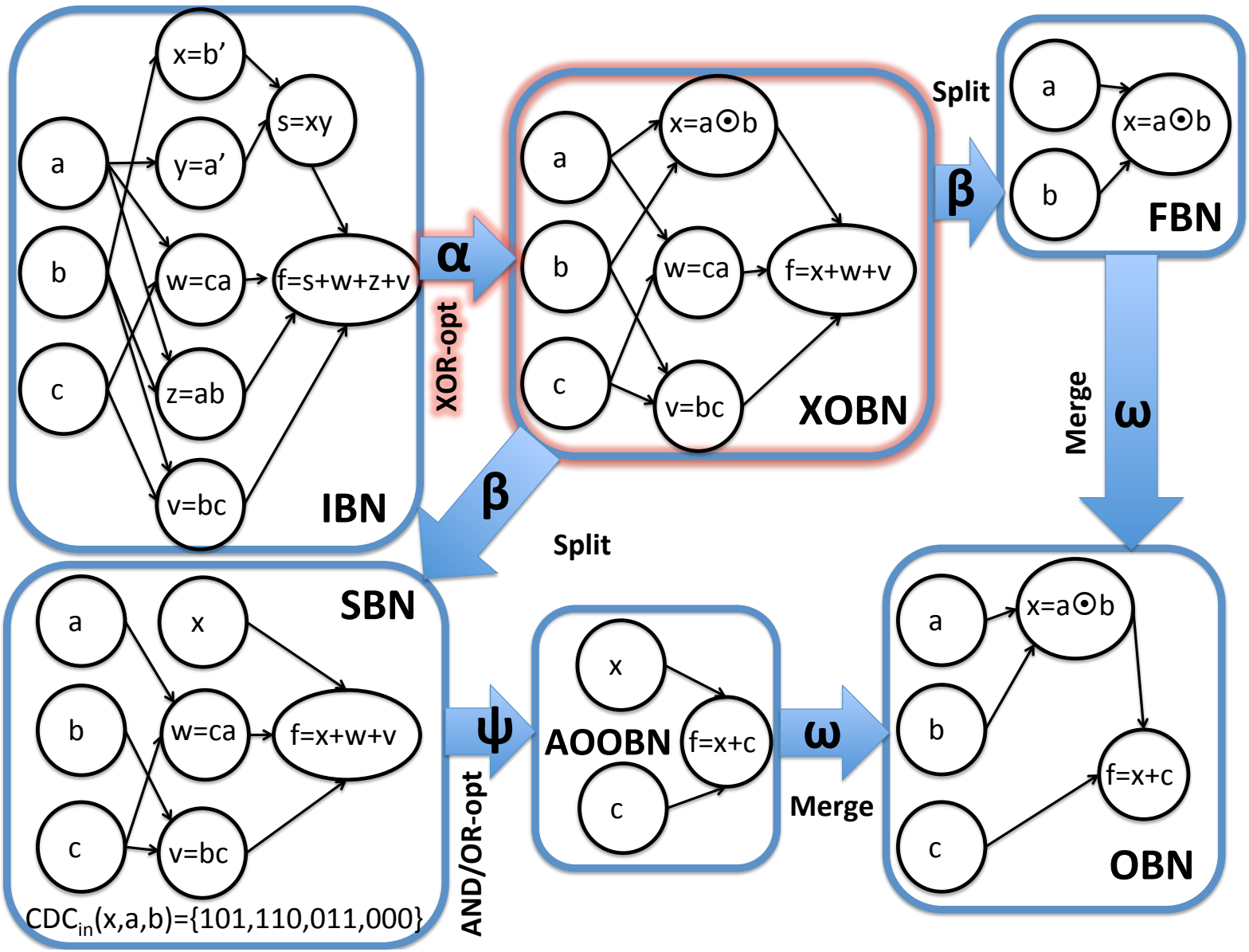
$$f = ab + bc + \overline{a}\overline{b} + ca$$



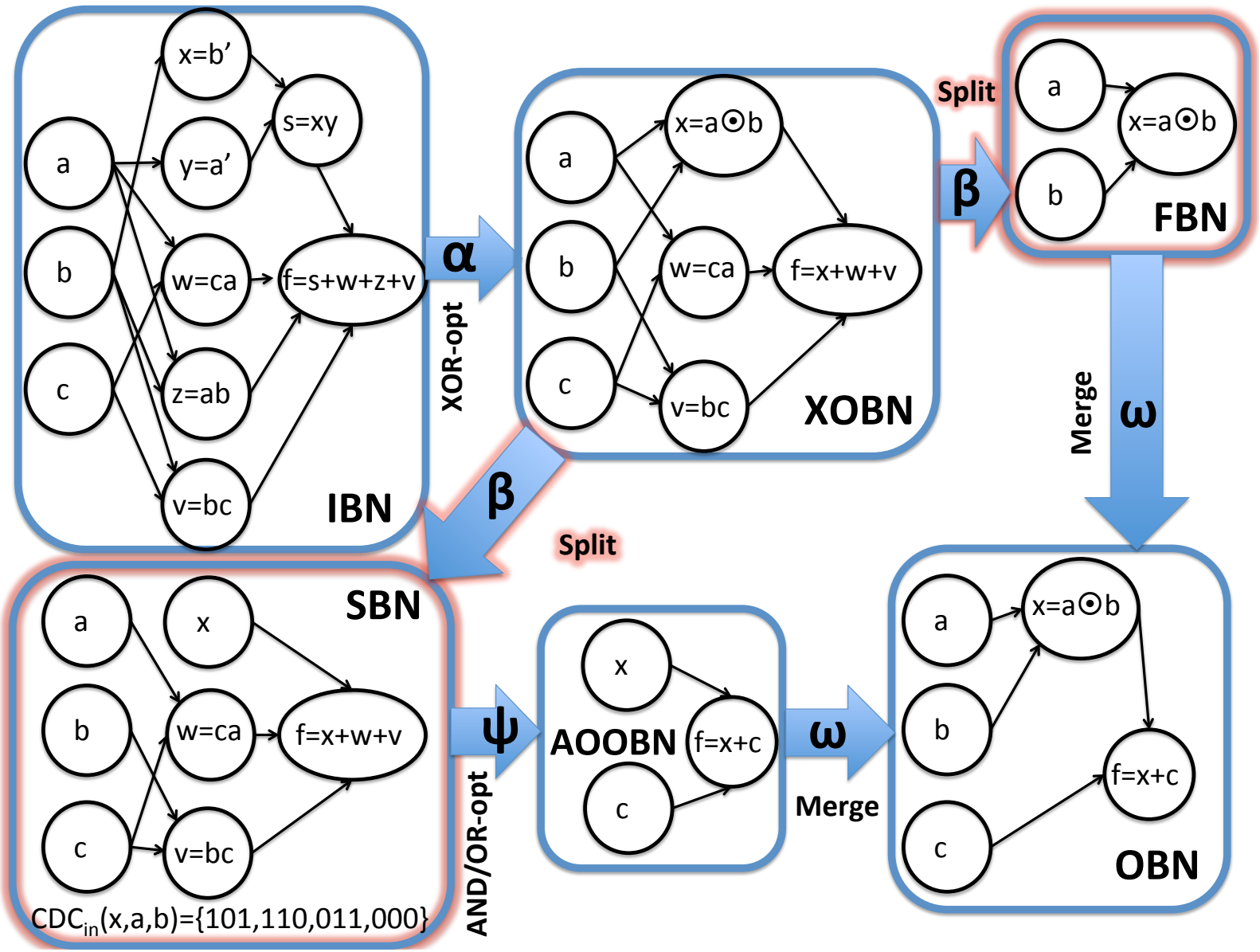
Hybrid Logic Optimization: Example



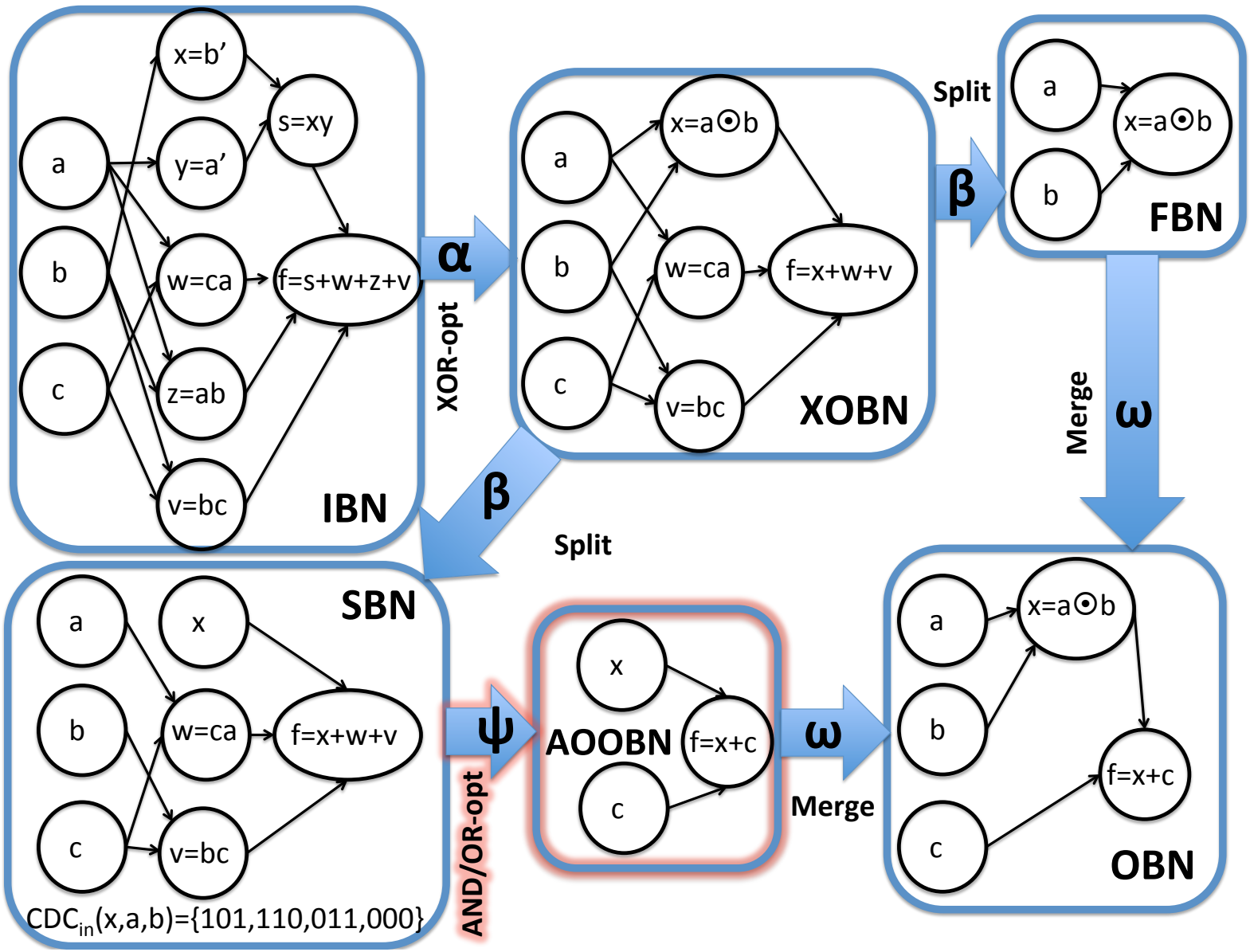
Hybrid Logic Optimization: Example



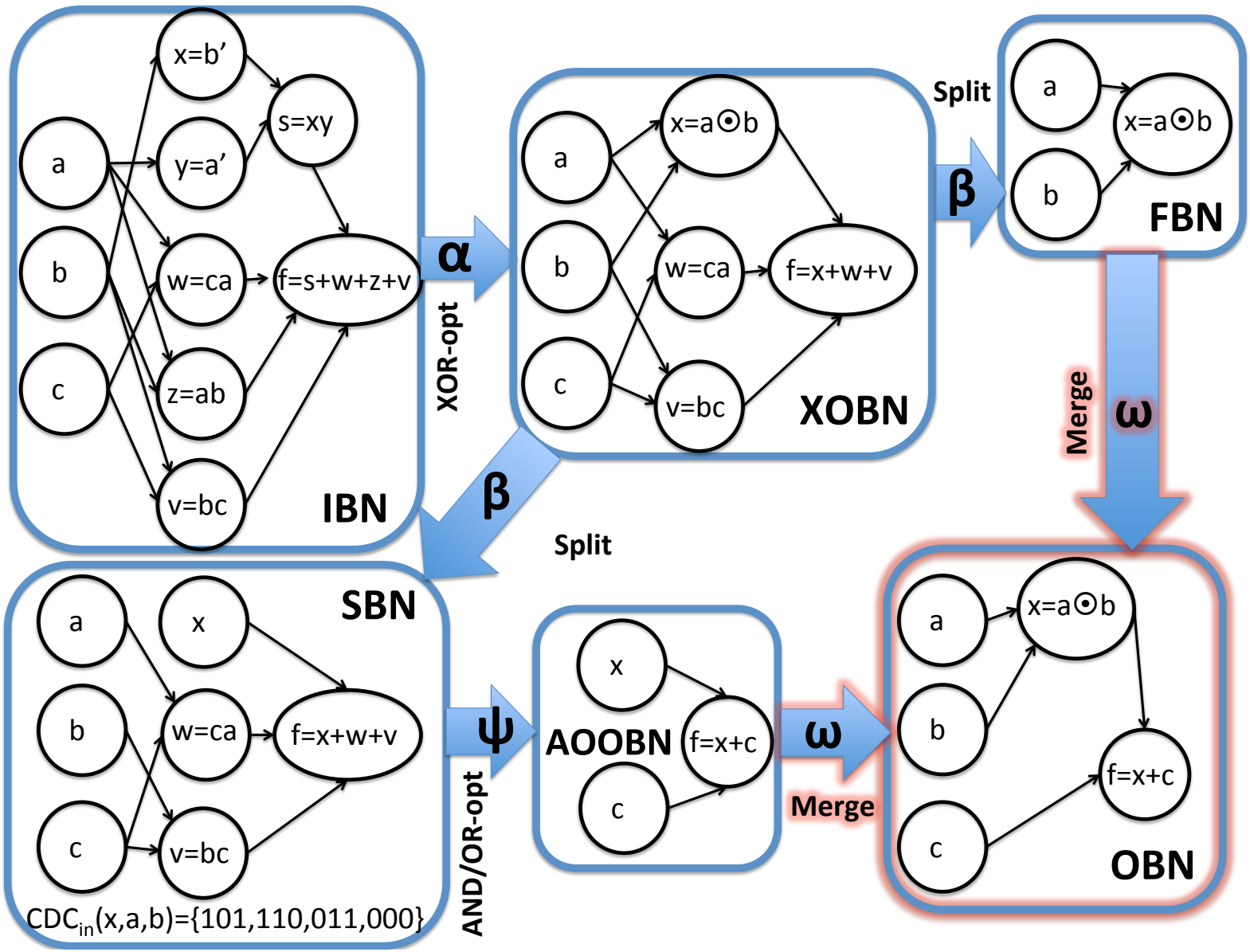
Hybrid Logic Optimization: Example



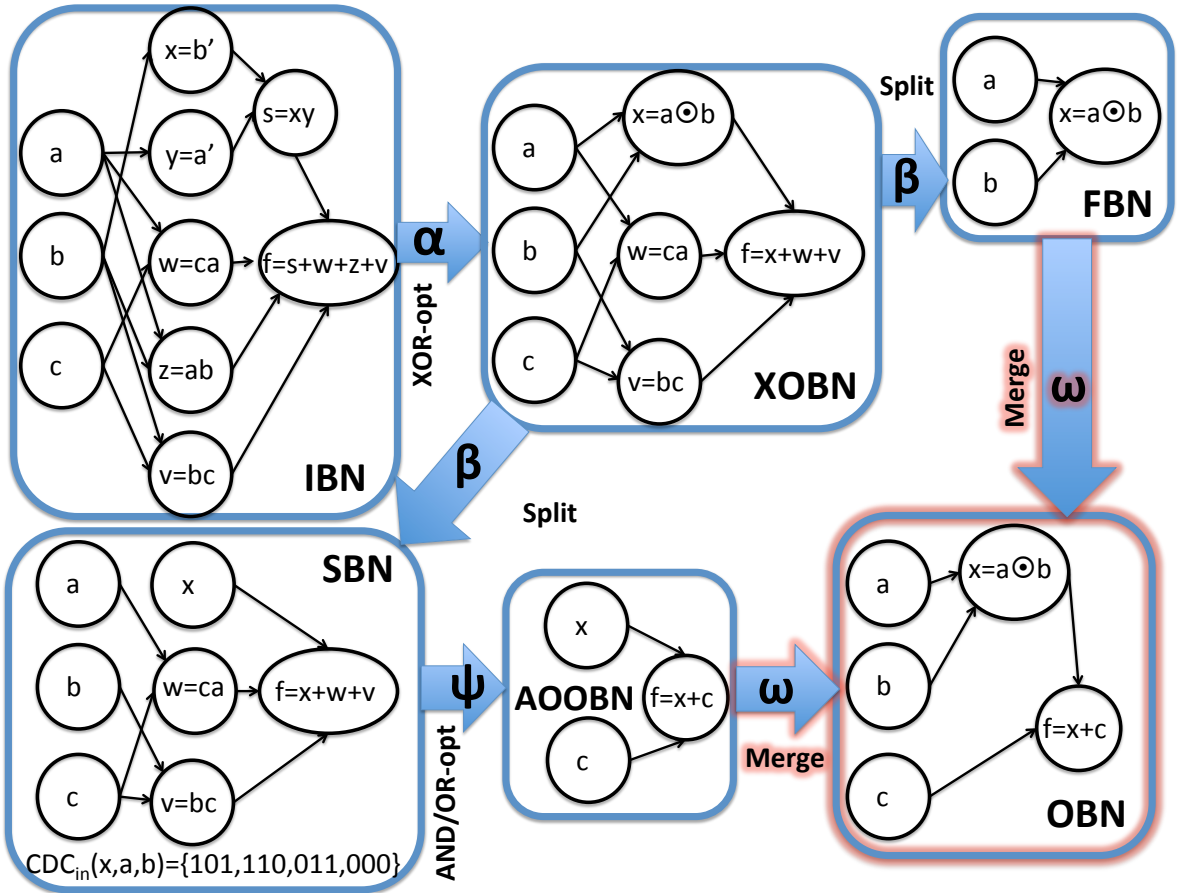
Hybrid Logic Optimization: Example



Hybrid Logic Optimization: Example



Hybrid Logic Optimization: Example



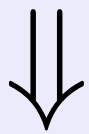
$$f = (a \odot b) + c$$

Minimal representation

MIXSyn Steps

MIXSyn

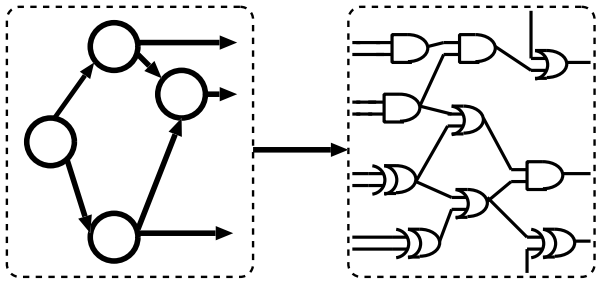
Hybrid optimization



Library-free technology mapping

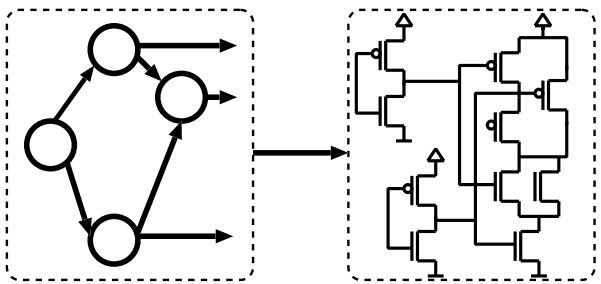
Library-free Technology Mapping: Opportunity

- **Conventional approach**: Standard cell based technology mapping.



  - Bounded by the richness of the given library.

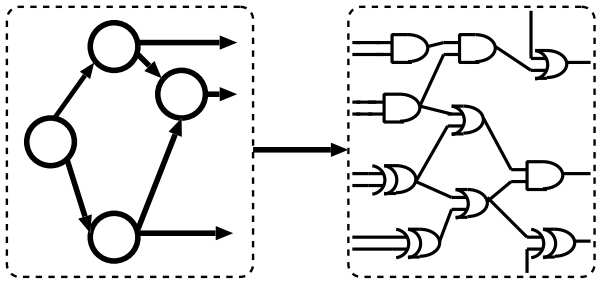
- **Novel opportunity**: Build on the fly logic gates at transistor level.



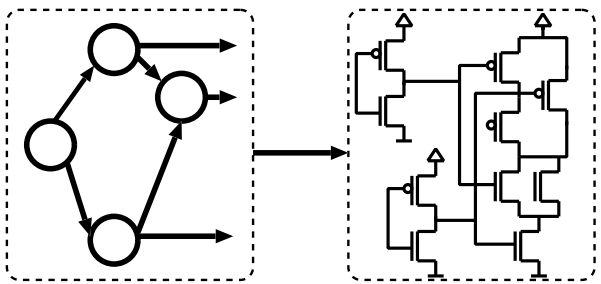
  - Implement the most convenient function: No more library bounds.
  - Exploit non-traditional gates: XOR-AND/OR expressions.
  - Selectively insert inverters: Maximize logic sharing.
  - Challenges for physical synthesis and timing characterization.

Library-free Technology Mapping: Opportunity

- **Conventional approach**: Standard cell based technology mapping.



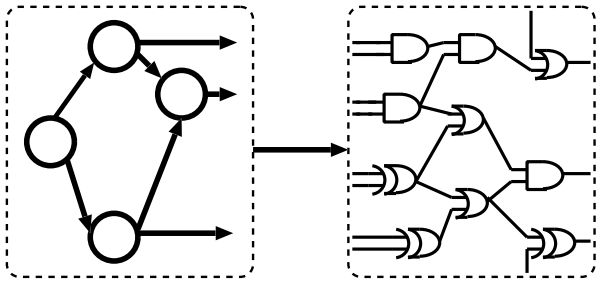
  - Bounded by the richness of the given library.
- **Novel opportunity**: Build on the fly logic gates at transistor level.



  - Implement the most convenient function: No more library bounds.
  - Exploit non-traditional gates: XOR-AND/OR expressions.
  - Selectively insert inverters: Maximize logic sharing.
  - Challenges for physical synthesis and timing characterization.

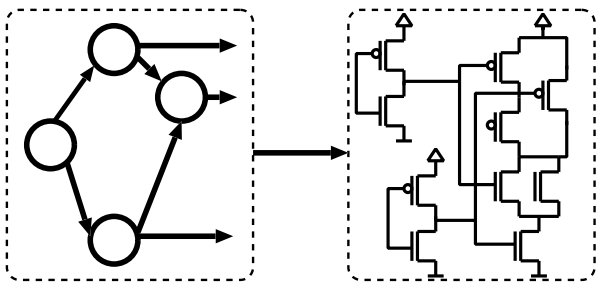
Library-free Technology Mapping: Opportunity

- **Conventional approach**: Standard cell based technology mapping.



  - Bounded by the richness of the given library.

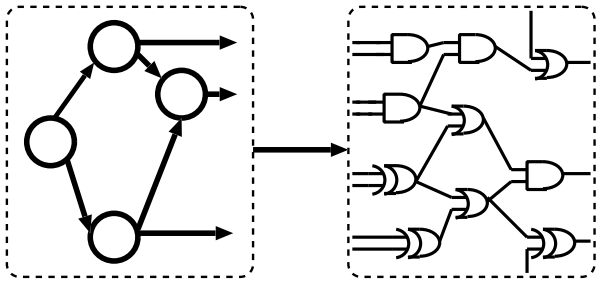
- **Novel opportunity**: Build on the fly logic gates at transistor level.



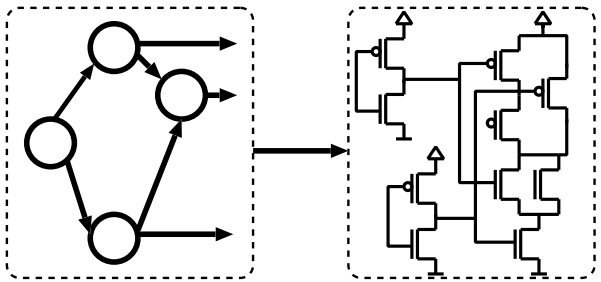
  - Implement the most convenient function: No more library bounds.
  - Exploit non-traditional gates: XOR-AND/OR expressions.
  - Selectively insert inverters: Maximize logic sharing.
  - Challenges for physical synthesis and timing characterization.

Library-free Technology Mapping: Opportunity

- **Conventional approach**: Standard cell based technology mapping.



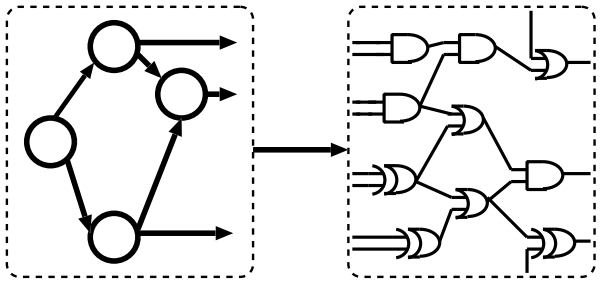
  - Bounded by the richness of the given library.
- **Novel opportunity**: Build on the fly logic gates at transistor level.



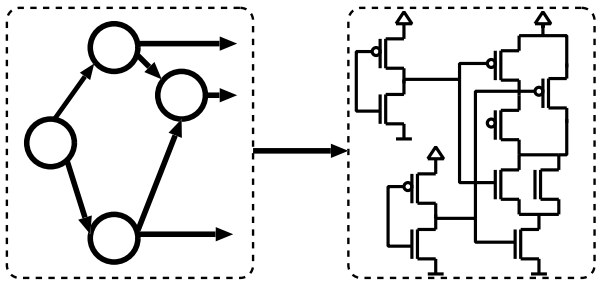
  - Implement the most convenient function: No more library bounds.
  - Exploit non-traditional gates: XOR-AND/OR expressions.
  - Selectively insert inverters: Maximize logic sharing.
  - Challenges for physical synthesis and timing characterization.

Library-free Technology Mapping: Opportunity

- **Conventional approach**: Standard cell based technology mapping.

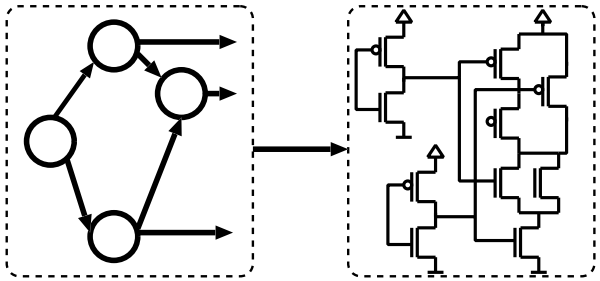


  - Bounded by the richness of the given library.
- **Novel opportunity**: Build on the fly logic gates at transistor level.



  - Implement the most convenient function: No more library bounds.
  - Exploit non-traditional gates: XOR-AND/OR expressions.
  - Selectively insert inverters: Maximize logic sharing.
  - Challenges for physical synthesis and timing characterization.

Library-free Technology Mapping: Procedure



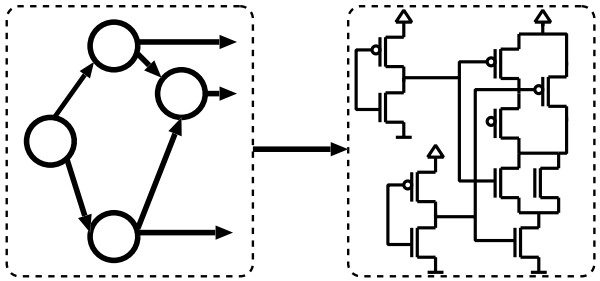
Procedure steps:

Algorithm 7 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
  Inverter propagation (phase-ω).
  Best polarity evaluation (phase-ω).
  Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



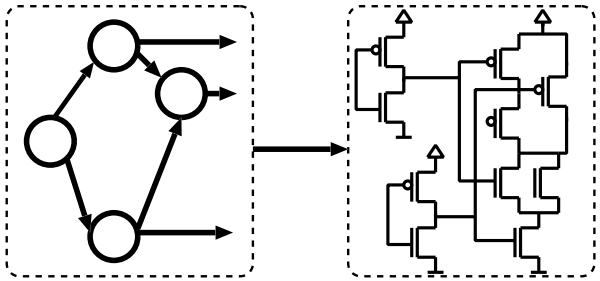
Procedure steps:

Algorithm 8 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
Inverter propagation (phase-ω).
Best polarity evaluation (phase-ω).
Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



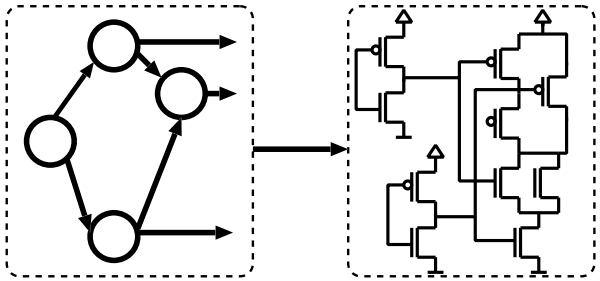
Procedure steps:

Algorithm 9 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
Inverter propagation (phase-ω).
Best polarity evaluation (phase-ω).
Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



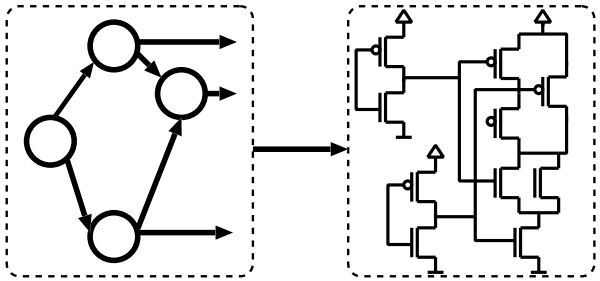
Procedure steps:

Algorithm 10 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
  Inverter propagation (phase-ω).
  Best polarity evaluation (phase-ω).
  Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



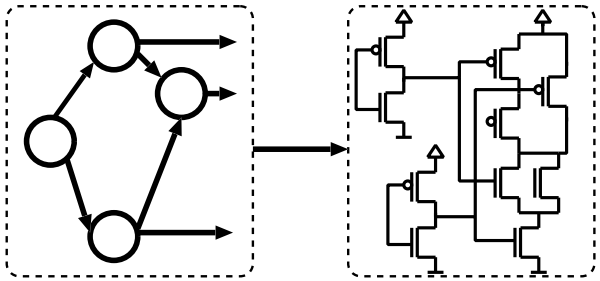
Procedure steps:

Algorithm 11 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
Inverter propagation (phase-ω).
Best polarity evaluation (phase-ω).
Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



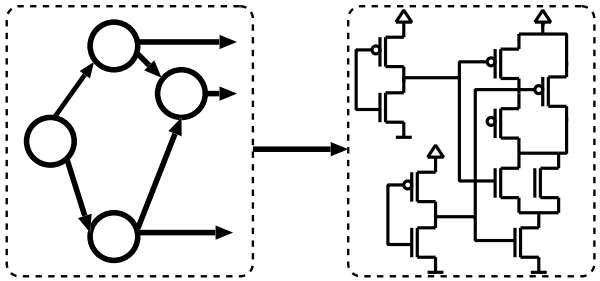
Procedure steps:

Algorithm 12 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
  Inverter propagation (phase-ω).
  Best polarity evaluation (phase-ω).
  Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



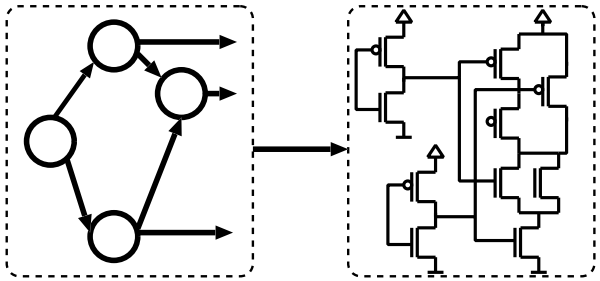
Procedure steps:

Algorithm 13 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
  Inverter propagation (phase-ω).
  Best polarity evaluation (phase-ω).
  Gate building (phase-ω).
end for

Library-free Technology Mapping: Procedure



Procedure steps:

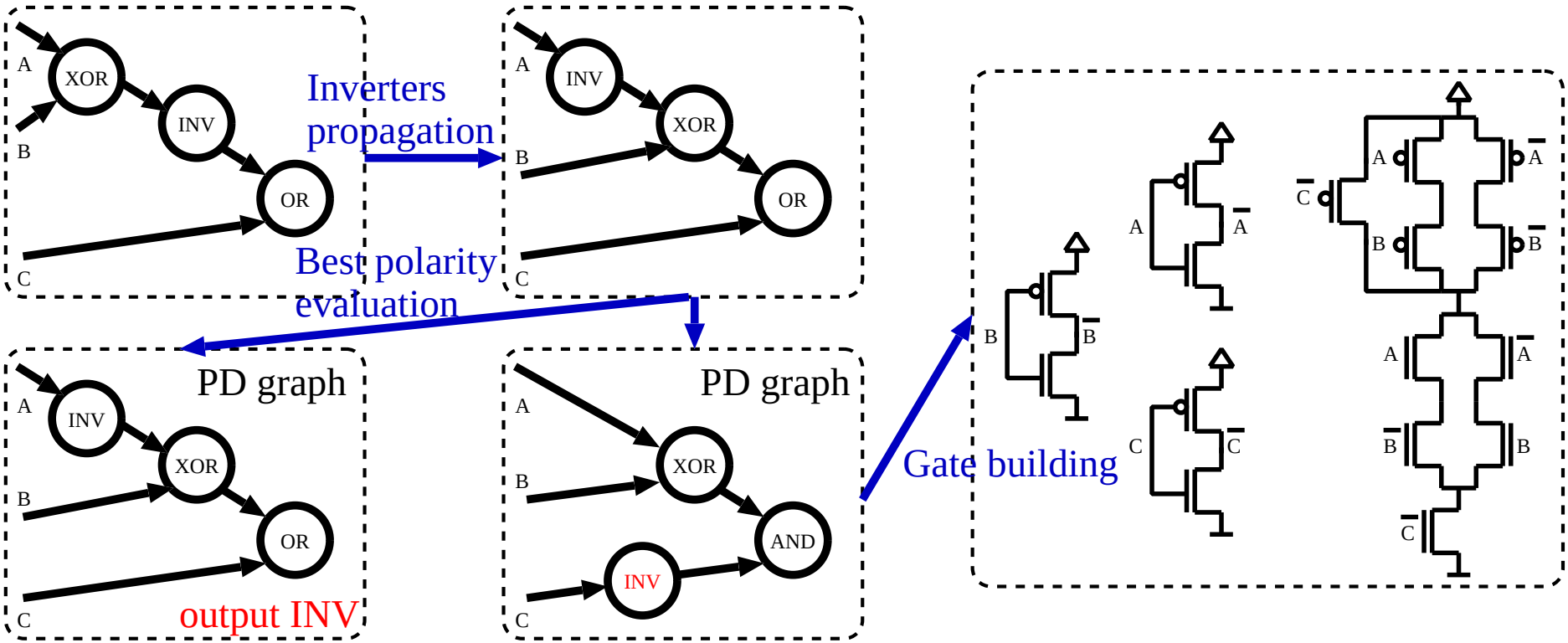
Algorithm 14 Library-free Technology Mapping

INPUT: Optimized Boolean Network (OBN), maximum gate fan-in
OUTPUT: Network of gates

Subject-graph creation (AND, OR, XOR, INV) (phase-α).
Forest of trees decomposition (phase-α).
Area efficient greedy tree decomposition (phase-β).
for all decomposed sub-trees **do**
  Inverter propagation (phase-ω).
  Best polarity evaluation (phase-ω).
  Gate building (phase-ω).
end for

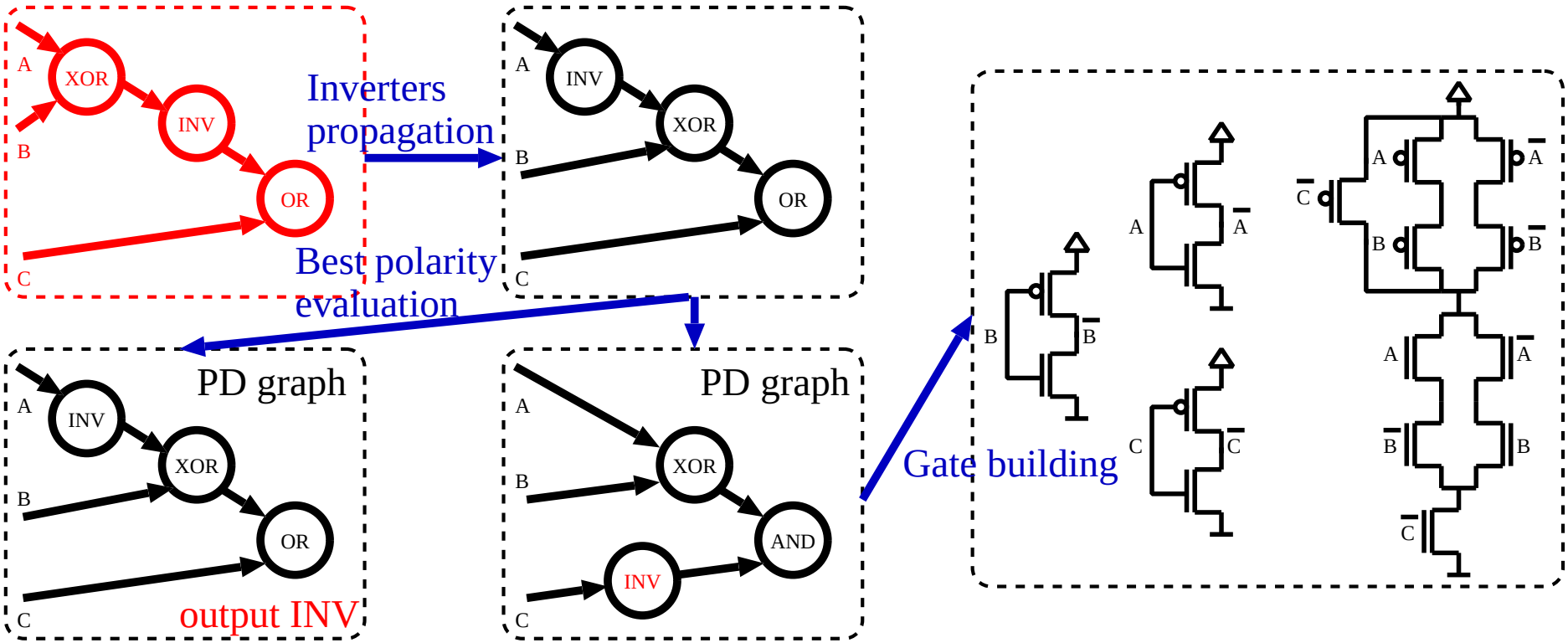
Library-free Technology Mapping: Example

$$f = (a \odot b) + c$$



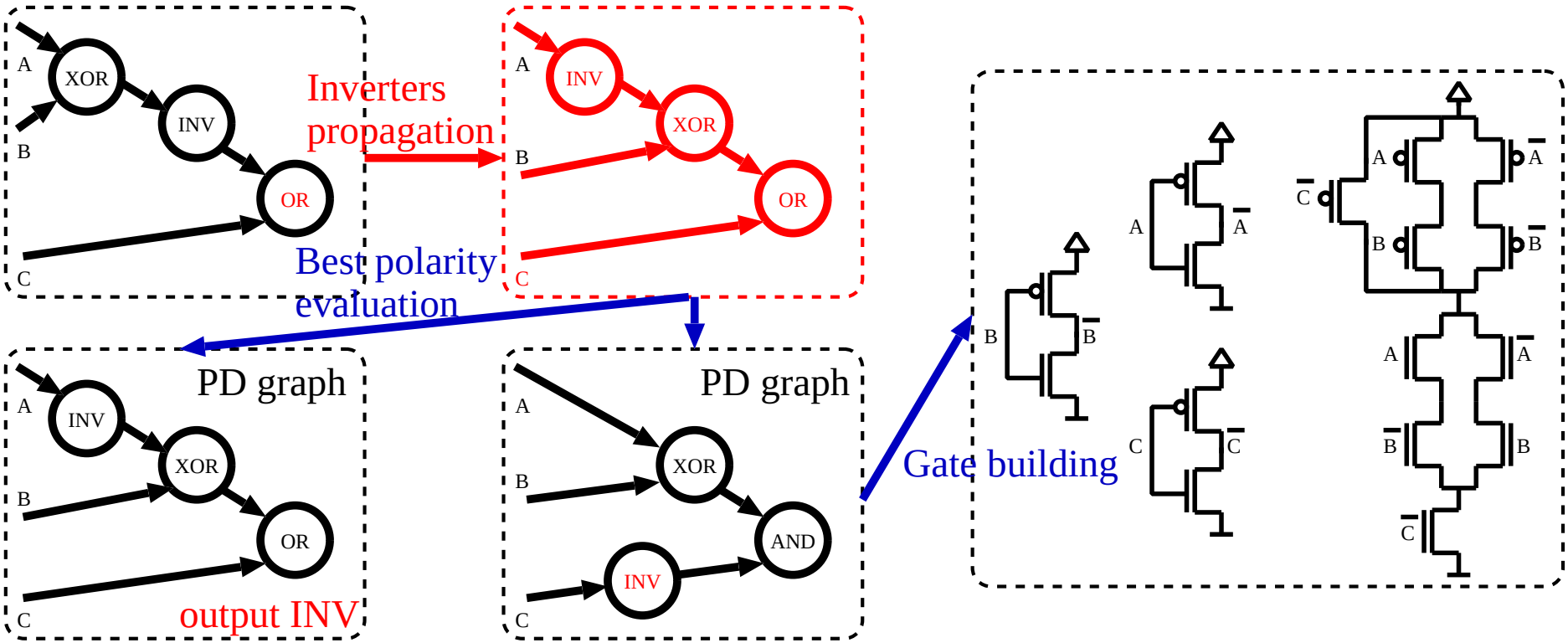
Library-free Technology Mapping: Example

$$f = (a \odot b) + c$$



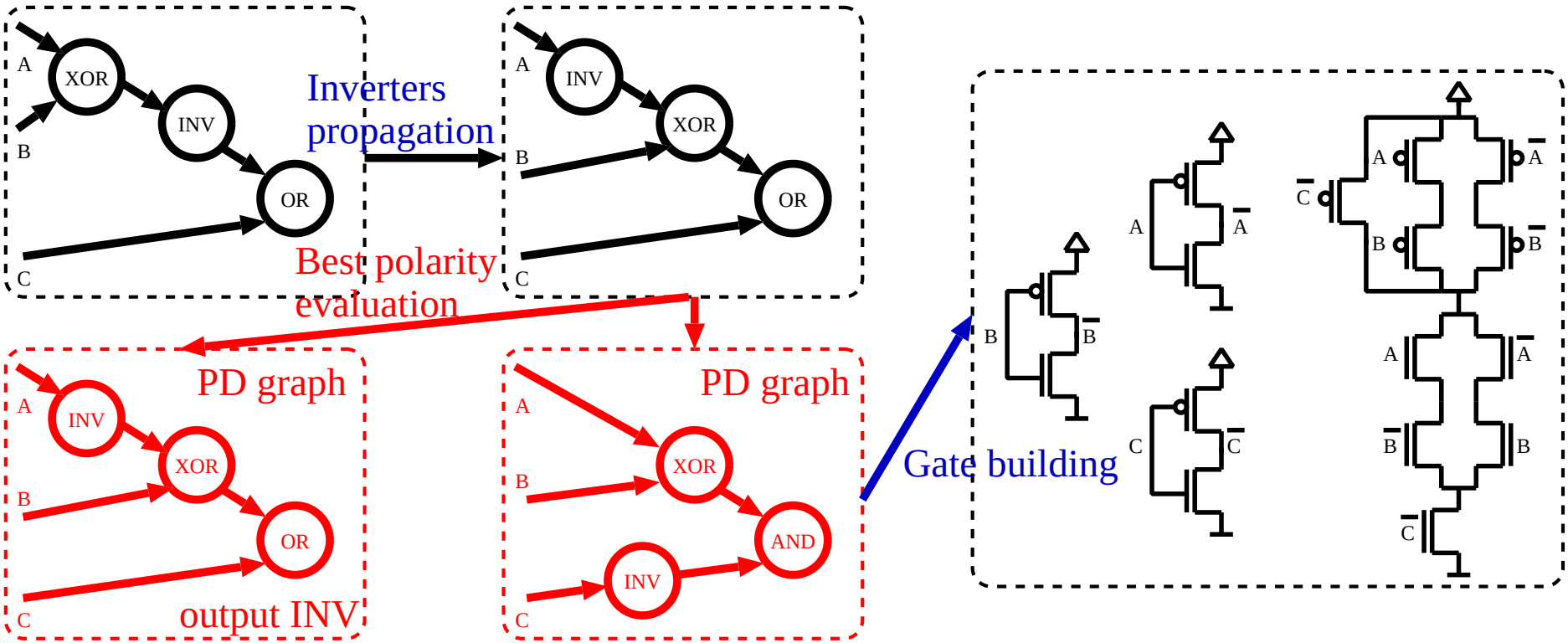
Library-free Technology Mapping: Example

$$f = (a \odot b) + c$$



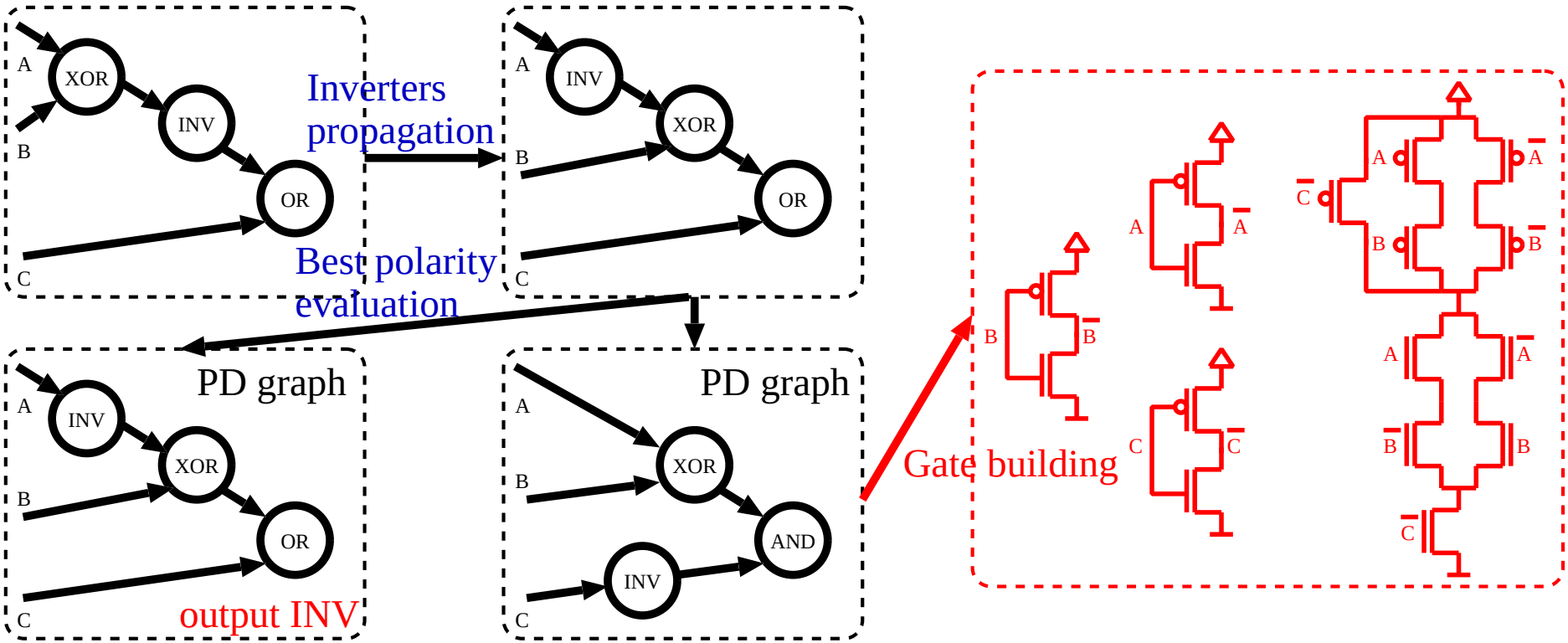
Library-free Technology Mapping: Example

$$f = (a \odot b) + c$$



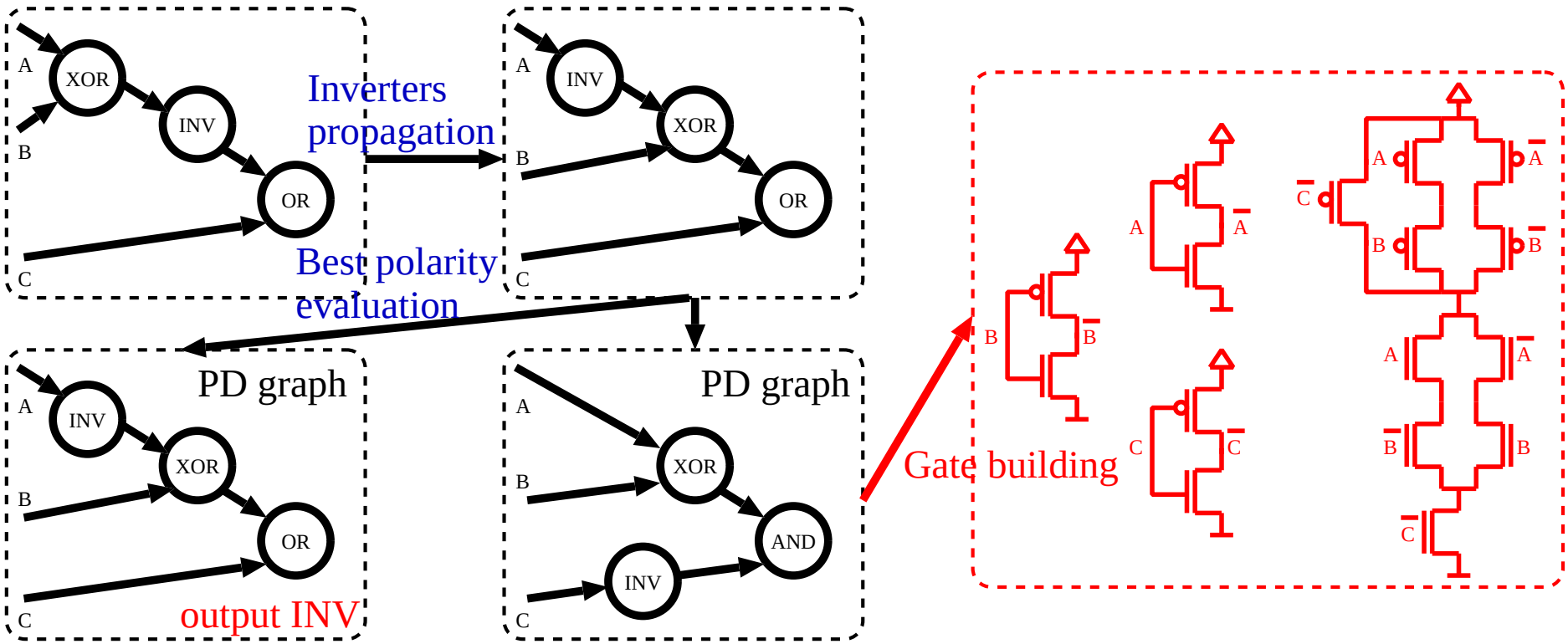
Library-free Technology Mapping: Example

$$f = (a \odot b) + c$$



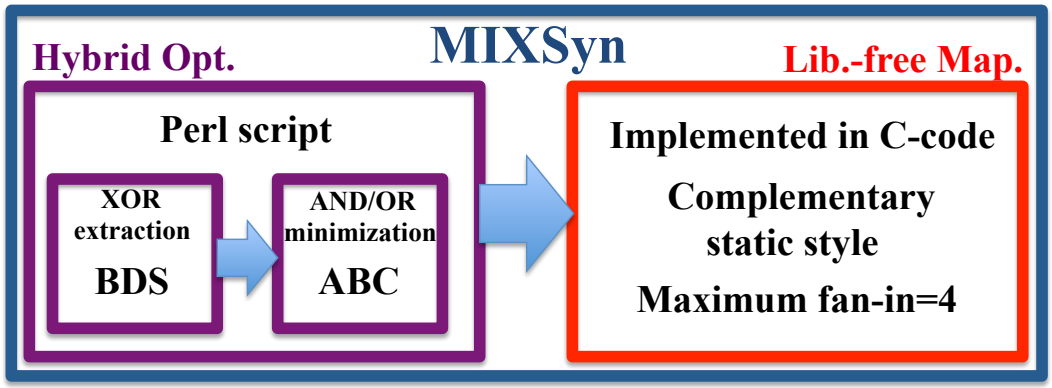
Library-free Technology Mapping: Example

$$f = (a \odot b) + c$$

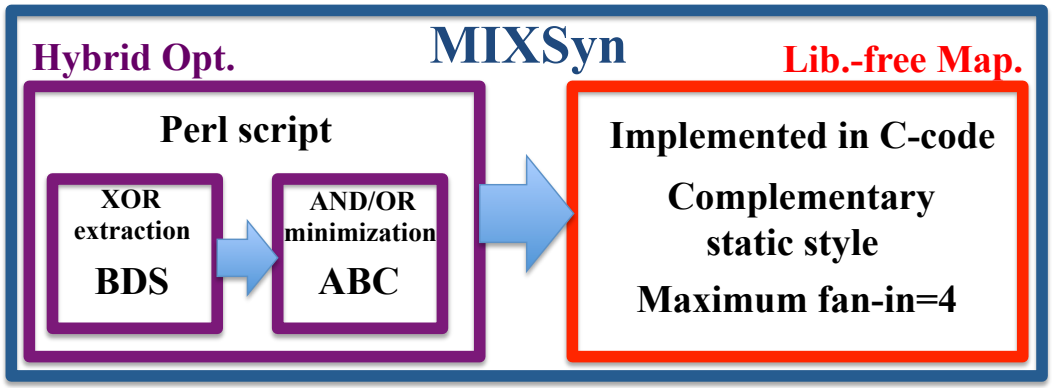


Minimum transistor count realization in static style.

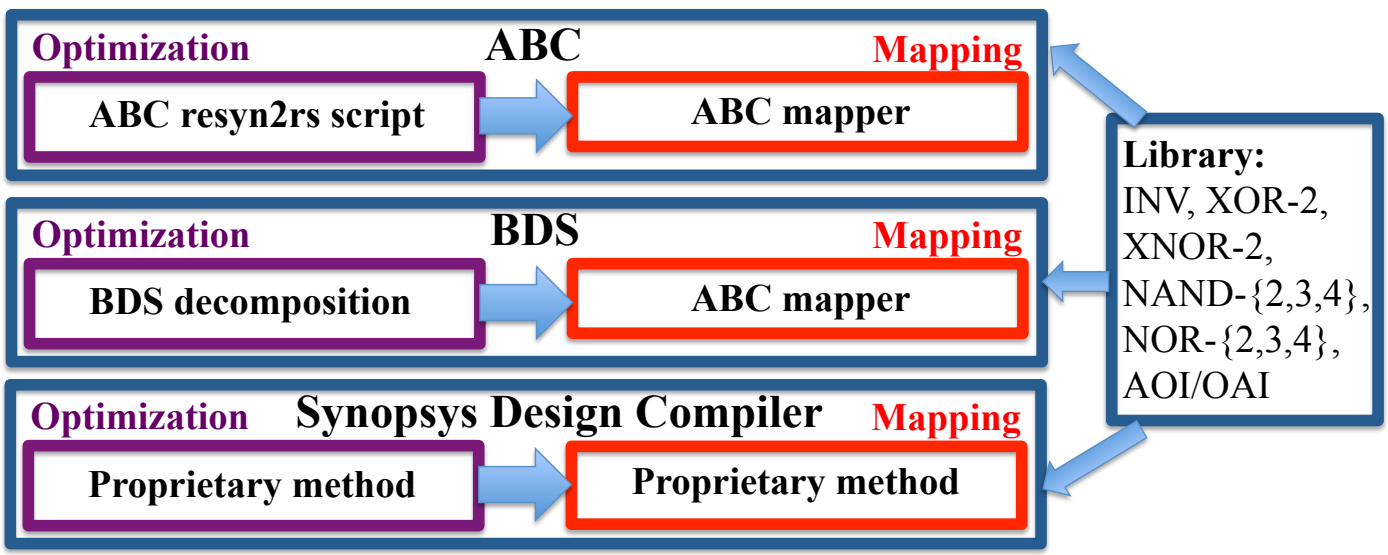
Experimental Setup



Experimental Setup

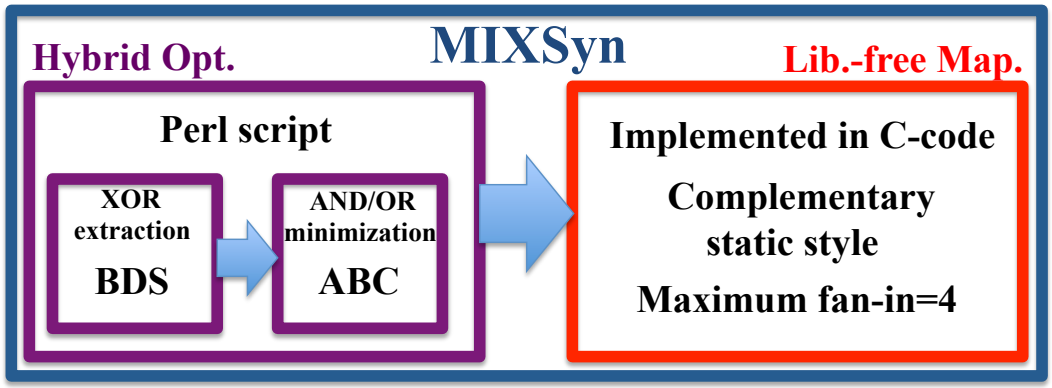


State of the art tools we compare with:

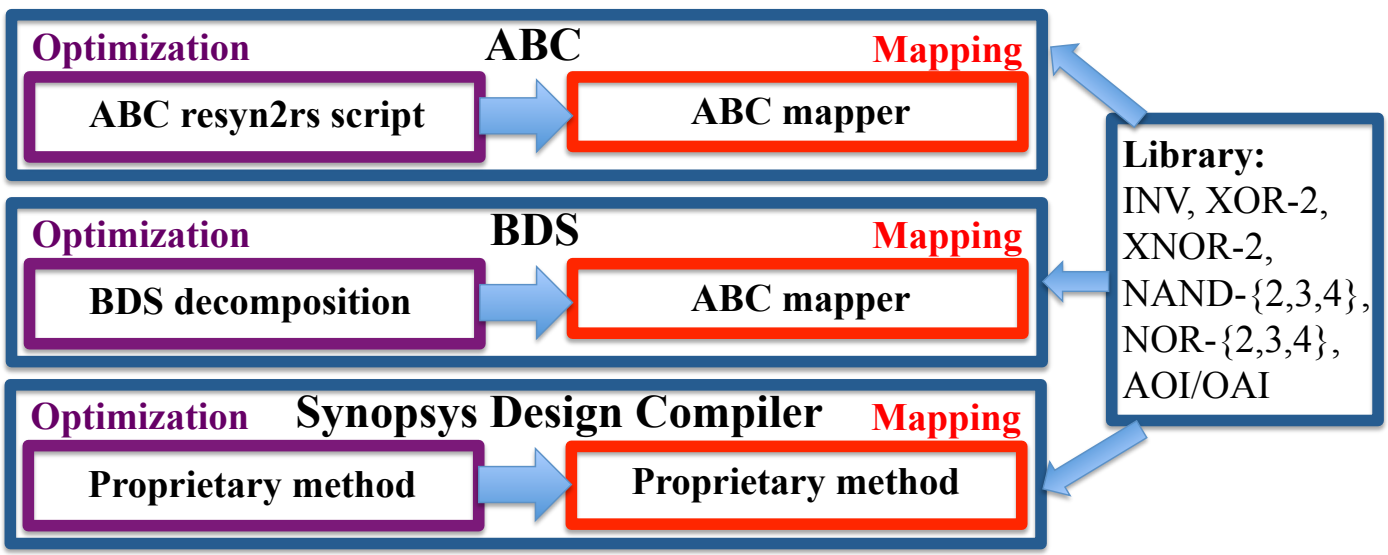


- Logic circuit benchmarks: MCNC suite.
- Comparison metric: **Transistor count** (area-efficiency).

Experimental Setup

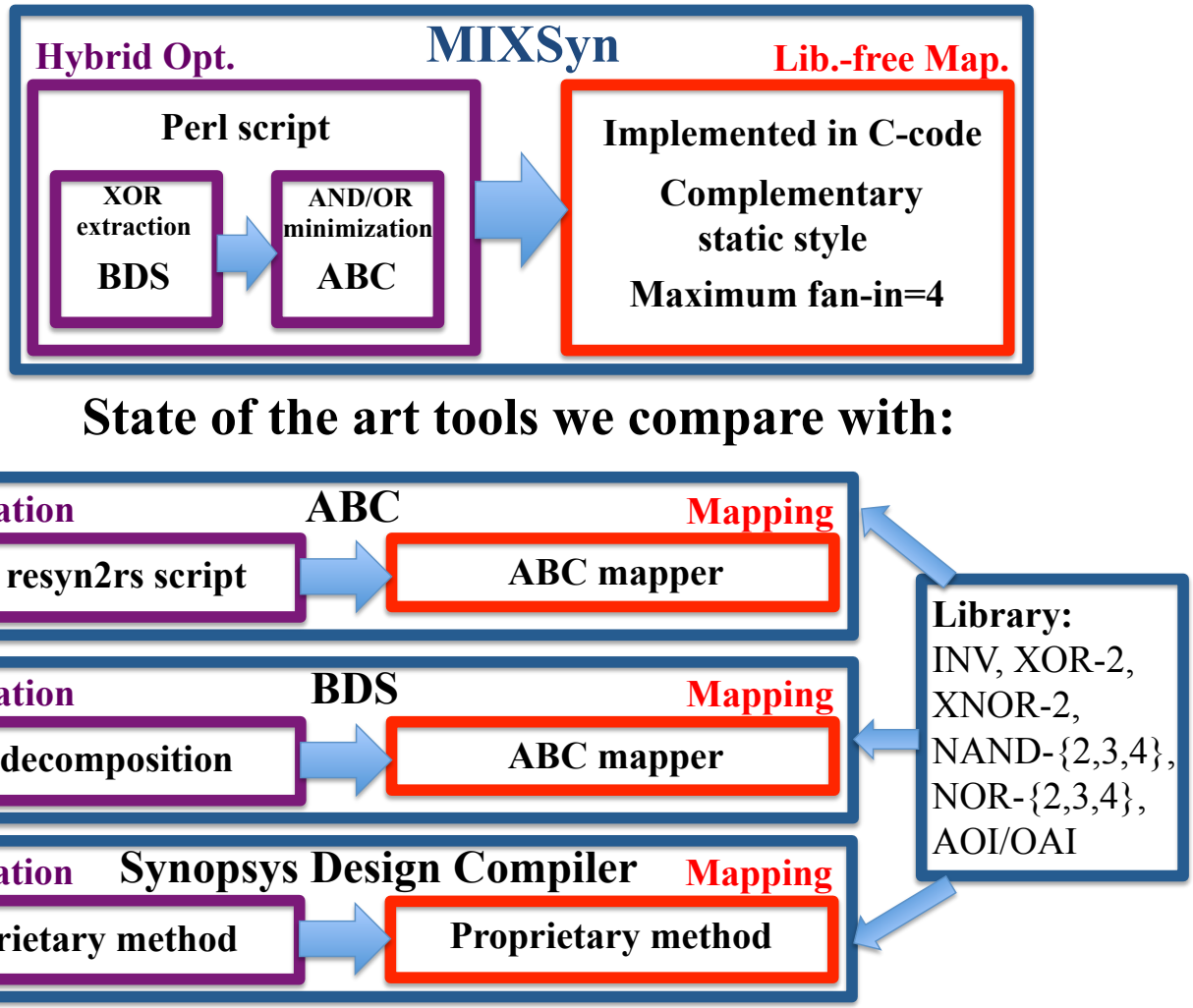


State of the art tools we compare with:



- Logic circuit benchmarks: MCNC suite.
- Comparison metric: **Transistor count** (area-efficiency).

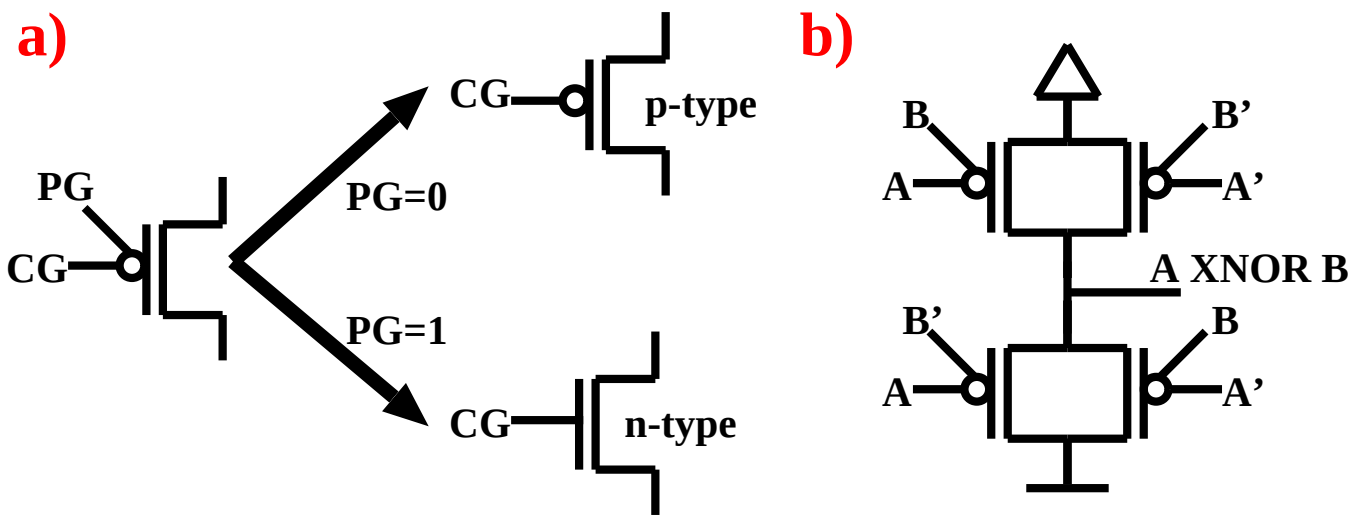
Experimental Setup



- Logic circuit benchmarks: MCNC suite.
- Comparison metric: **Transistor count** (area-efficiency).

Emerging XOR-efficient Devices

- We benchmark MIXSyn with:
- **[1]** Standard CMOS.
- **[2]** Emerging XOR-efficient device: **double-gate ambipolar FETs**.
  - Electrically controllable polarity trough polarity gate (PG).
  - PG=1 $\Rightarrow$ n-type, PG=0 $\Rightarrow$ p-type.
  - Fabricated in SiNW[1], graphene and carbon nanotubes.



  - *Biconditional* logic connective embedded (XNOR $\odot$).
  - Compact **XOR/XNOR**-based logic gates.

1: De Marchi *et al.*, IEDM 2012

CMOS Synthesis Experiments

Target technology: CMOS

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
Full Set	4931	5433	6016	6063

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	4129	4174	4749	4959

CMOS Synthesis Experiments

Target technology: CMOS

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4931**	5433	6016	6063

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**4129**	4174	4749	4959

CMOS Synthesis Experiments

Target technology: CMOS

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4931**	**5433**	**6016**	**6063**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**4129**	4174	4749	4959

CMOS Synthesis Experiments

Target technology: CMOS

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4931**	**+10.2%**	**+22.0%**	**+23.0%**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**4129**	4174	4749	4959

CMOS Synthesis Experiments

Target technology: CMOS

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4931**	**+10.2%**	**+22.0%**	**+23.0%**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**4129**	**4174**	**4749**	**4959**

CMOS Synthesis Experiments

Target technology: CMOS

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4931**	**+10.2%**	**+22.0%**	**+23.0%**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**4129**	**+1.1%**	**+15.0%**	**+20.1%**

Ambipolar Synthesis Experiments

Target technology: DG Ambipolar FETs

Tools ranking (transistor count)

Benchmarks	MIXSyn (1st)	DC (2nd)	ABC (3rd)	BDS (4th)
Full Set	4204	4965	5312	5346

Benchmarks	MIXSyn (1st)	DC (2nd)	BDS (3rd)	ABC (4th)
XOR-int.	3136	3603	3944	3958

Ambipolar Synthesis Experiments

Target technology: DG Ambipolar FETs

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4204**	4965	5312	5346

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**3136**	3603	3944	3958

Ambipolar Synthesis Experiments

Target technology: DG Ambipolar FETs

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4204**	**4965**	**5312**	**5346**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**3136**	3603	3944	3958

Ambipolar Synthesis Experiments

Target technology: DG Ambipolar FETs

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4204**	**+18.1%**	**+26.3%**	**+27.2%**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**3136**	3603	3944	3958

Ambipolar Synthesis Experiments

Target technology: DG Ambipolar FETs

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4204**	**+18.1%**	**+26.3%**	**+27.2%**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**3136**	**3603**	**3944**	**3958**

Ambipolar Synthesis Experiments

Target technology: DG Ambipolar FETs

Tools ranking (transistor count)

Benchmarks	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Full Set	**4204**	**+18.1%**	**+26.3%**	**+27.2%**

Benchmarks	MIXSyn	DC	BDS	ABC
	(1st)	(2nd)	(3rd)	(4th)
XOR-int.	**3136**	**+14.9%**	**+25.7%**	**+26.2%**

Ambipolar vs. CMOS Synthesis Experiments

MIXSyn further harnesses ambipolar technology.

Tools ranking (transistor count)

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	4931	5433	6063	6016
Ambipolar	4204	4965	5312	5346

Ambipolar vs. CMOS Synthesis Experiments

MIXSyn further harnesses ambipolar technology.

Tools ranking (transistor count)

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	4931	5433	6063	6016
Ambipolar	**4204**	4965	5312	5346

Ambipolar vs. CMOS Synthesis Experiments

MIXSyn further harnesses ambipolar technology.

Tools ranking (transistor count)

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	4931	5433	6063	6016
Ambipolar	**-14.7%**	4965	5312	5346

Ambipolar vs. CMOS Synthesis Experiments

MIXSyn further harnesses ambipolar technology.

Tools ranking (transistor count)

Technology	**MIXSyn**	DC	**ABC**	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	4931	5433	6063	6016
Ambipolar	**-14.7%**	**4965**	**5312**	**5346**

Ambipolar vs. CMOS Synthesis Experiments

MIXSyn further harnesses ambipolar technology.

Tools ranking (transistor count)

Technology	**MIXSyn**	DC	**ABC**	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	4931	5433	6063	6016
Ambipolar	**-14.7%**	**-8.6%**	**-12.4%**	**-11.1%**

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	33.2	34.5	36.7	38.1

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	34.5	36.7	38.1

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	34.5	36.7	38.1

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**34.5**	36.7	38.1

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	36.7	38.1

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	**36.7**	**38.1**

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	**+10.5%**	**+14.7%**

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	22.8	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	**+10.5%**	**+14.7%**

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	**22.8**	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	**+10.5%**	**+14.7%**

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	**22.8**	23.9	25.6	26.1

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	**+10.5%**	**+14.7%**

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	**22.8**	**23.9**	**25.6**	**26.1**

Timing: Logic Depth Results

Insight about timing efficiency

Tools ranking (logic levels)

Technology	DC	MIXSyn	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
CMOS	**33.2**	**+3.9%**	**+10.5%**	**+14.7%**

Technology	MIXSyn	DC	ABC	BDS
	(1st)	(2nd)	(3rd)	(4th)
Ambipolar	**22.8**	**+4.8%**	**+12.2%**	**+14.4%**

Conclusions

- We presented **MIXSyn**: A novel logic synthesis methodology.
- **MIXSyn** targets XOR-AND/OR dominated circuits.
- **MIXSyn**: hybrid optimization + library-free mapping.

- **MIXSyn** produces CMOS efficient circuits:
  - Transistor count: -10.2% w.r.t. Design Compiler.
  - Logic levels: +3.9% w.r.t. Design Compiler.
- **MIXSyn** harnesses DG ambipolar devices expressive power:
  - Transistor count: -18.1% w.r.t. Design Compiler.
  - Logic levels: -4.8% w.r.t. Design Compiler.

Questions?

Thank you for your attention.