

Reversible Denial-of-Service by Locking Gates Insertion for IP Cores Design Protection

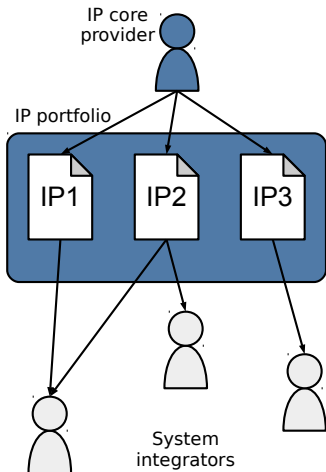
Brice COLOMBIER*, Lilian BOSSUET*, David HÉLY⁺

*Laboratoire Hubert Curien
Université Jean Monnet
Saint-Étienne — France

⁺LCIS
Grenoble Institute of Technology
Valence — France



Design-and-reuse paradigm



Problem

The designer cannot control **how many times** the IP is instantiated.

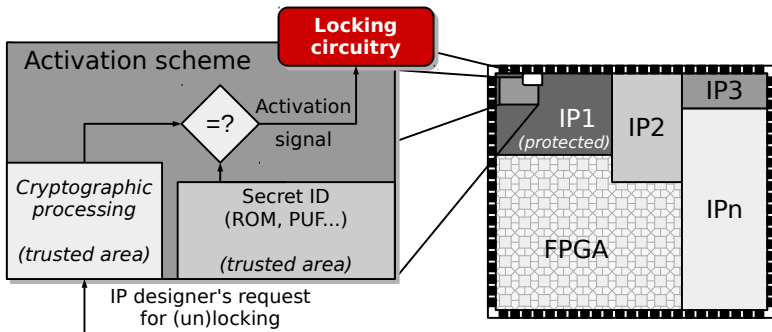
- Overusing,
- Illegal cloning.

One solution

Make the IP unusable unless it has been previously **activated**.

⇒ Illegal copies are useless.

Protection scheme architecture



Each has its role:

Security : relies on a **cryptographic primitive**.

Uniqueness : ID, NVM, PUF...

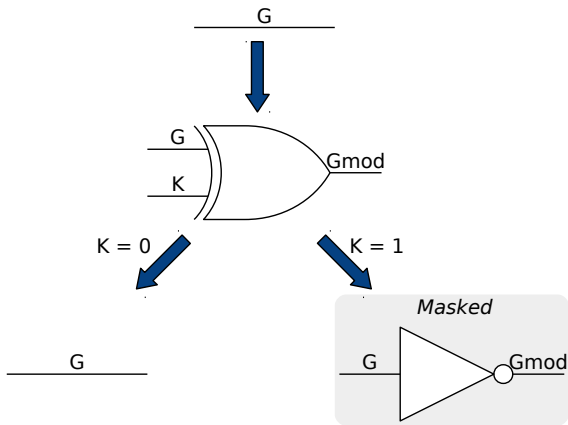
Disabling : specific masking/**locking** module.

Locking

- FSM,
- Clock,
- **Logic**.

Logic masking

In 2008, Roy et al.¹ proposed to **randomly** insert XOR/XNOR gates in the netlist.



¹ Roy, Koushanfar, Markov EPIC: Ending Piracy of Integrated Circuits
Design, Automation and Test in Europe, 2008

Logic masking

In 2013, Rajendran et al.² improved the **node selection** method.

Fault analysis-based node selection

- Requires a fault simulator,
- Computationally expensive.

² **Rajendran, Zhang, Rose, Pino, Sinanoglu, Karri** *Fault analysis-based logic encryption* IEEE Transactions on Computers, 2013

³ **Plaza, Markov** *Protecting Integrated Circuits from Piracy with Test-aware Logic Locking* International Conference on Computer Aided Design, 2014

Logic masking

In 2013, Rajendran et al.² improved the **node selection** method.

Fault analysis-based node selection

- Requires a fault simulator,
- Computationally expensive.

Security flaw

There is a **gradient** towards the correct key.

A **hill climbing** attack³ can be mounted:

Choose a random key, flip bits **one after the other** to gradually reduce $HD(output, test\ vectors)$ to 0.

Link between key bits and masked outputs is **too obvious**.

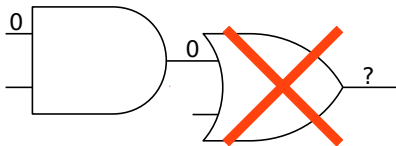
²Rajendran, Zhang, Rose, Pino, Sinanoglu, Karri *Fault analysis-based logic encryption* IEEE Transactions on Computers, 2013

³Plaza, Markov *Protecting Integrated Circuits from Piracy with Test-aware Logic Locking* International Conference on Computer Aided Design, 2014

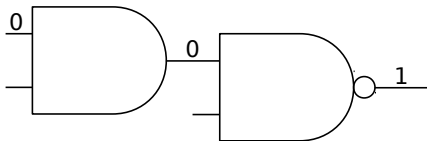
What is logic locking?



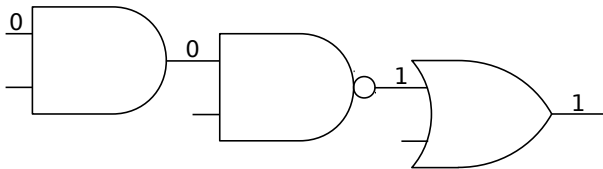
What is logic locking?



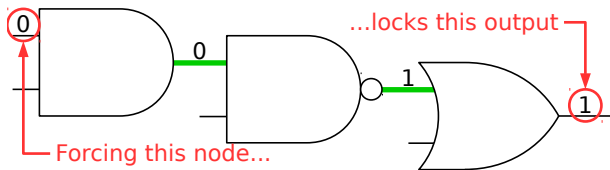
What is logic locking?



What is logic locking?



What is logic locking?



Principle

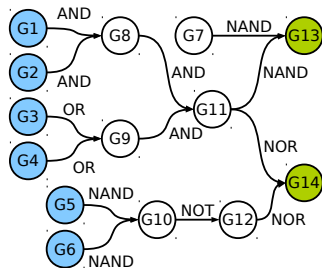
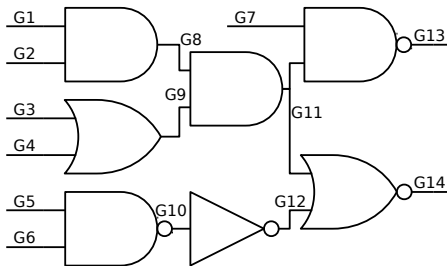
Propagating a **locking value** through a sequence of netlist's nodes. Forcing an internal node in the netlist **locks a primary output**.

Condition

For all the nodes in the sequence (green nodes):
They are **forced** to a logic value that **locks** the following logic gate.

Which node should be forced ?

1st step: Build a graph from the netlist.





Conversion

Nodes

→

Vertices

 Input

 Output

Gates

→

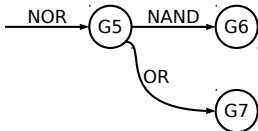
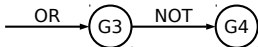
Edges

Graph labelling

2nd step: Label vertices with V_{forced} and V_{locks} values.

Labelling

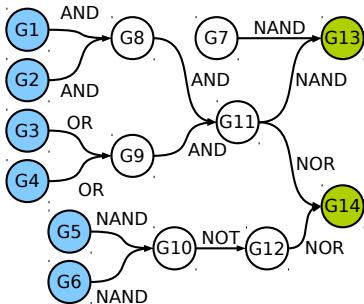
- V_{forced} depends on the **preceding** logic gate.
- V_{locks} depends on the **following** logic gate.



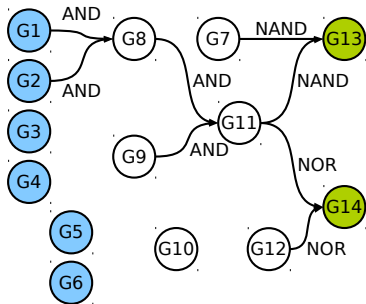
Node	V_{forced}	V_{locks}
G1	0	1
G2	0	0
G3	1	$\sim V_{locks}(G4)$
G5	0	{0, 1}

Graph simplification

3rd step: Delete **incoming edges** of nodes for which $V_{forced} \notin V_{locks}$. Those nodes **cannot propagate** the locking value.



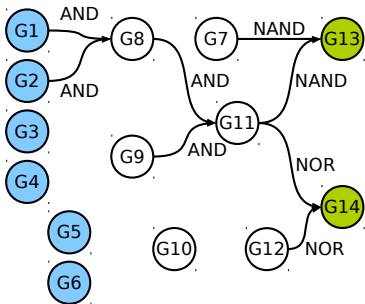
Original graph



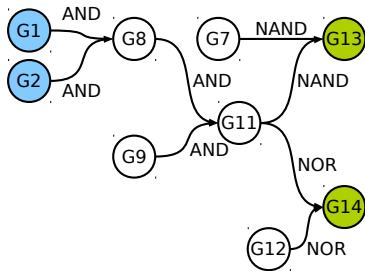
Simplified graph

Graph cleaning

4th step: Delete connected components that contain **no output**.



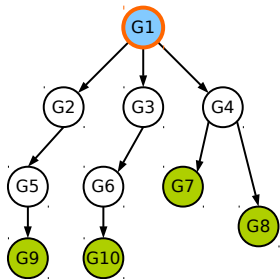
Simplified graph



Cleaned graph

Nodes selection

In the disconnected final graph, which nodes should be locked?

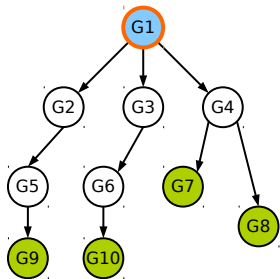


One source vertex

Select the source vertex.

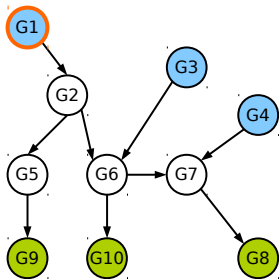
Nodes selection

In the disconnected final graph, which nodes should be locked?



One source vertex

Select the source vertex.

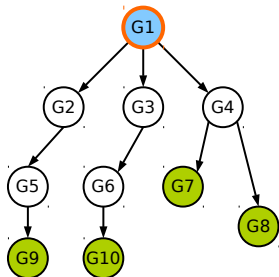


Multiple source vertices

Select the furthest node spanning all the outputs.

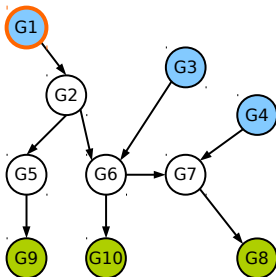
Nodes selection

In the disconnected final graph, which nodes should be locked?



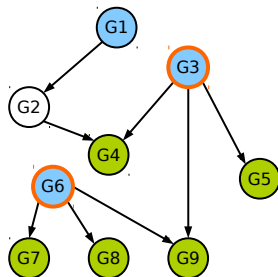
One source vertex

Select the source vertex.



Multiple source vertices

Select the furthest node spanning all the outputs.



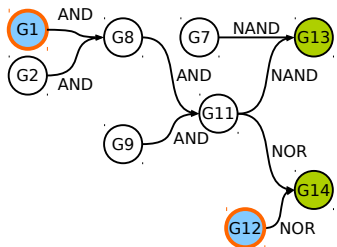
Multiple source vertices
not all outputs spanned

Sort the nodes w.r.t to the number of outputs they span and select them greedily.

Locking gates insertion

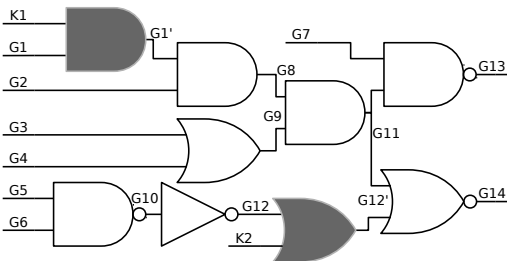
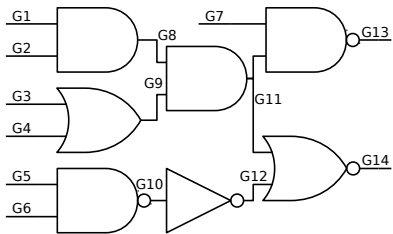
So far, we have:

- list of nodes to lock,
- associated V_{locks} values.



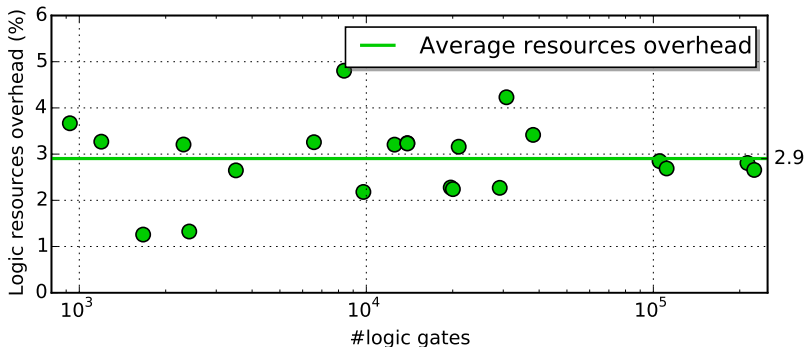
$V_{locks} = 0$: add AND gate

$V_{locks} = 1$: add OR gate



Area overhead

Overhead metric: percentage of locking gates to add.
Implemented on ITC'99 benchmarks (1k to 225k logic gates)

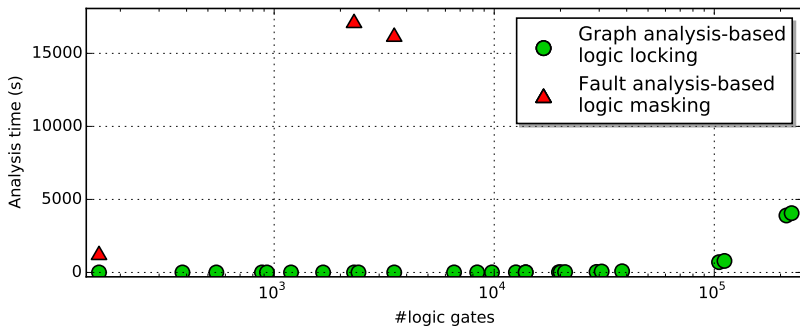


$\sim 2x$ lower overhead than logic masking² (+5.7%)

²Rajendran, Zhang, Rose, Pino, Sinanoglu, Karri *Fault analysis-based logic encryption* IEEE Transactions on Computers, 2013

Analysis time

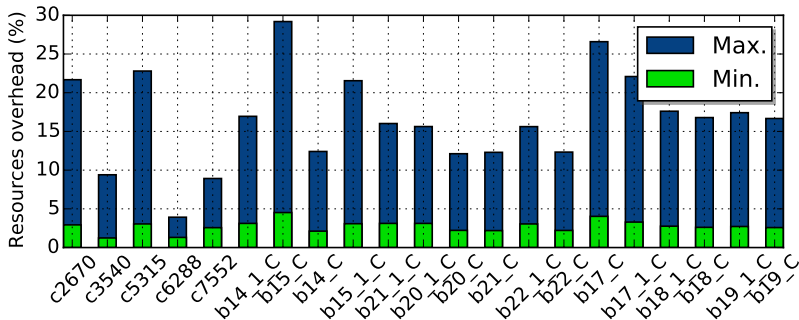
Benchmark	#logic gates	Fault analysis-based logic masking ²	Graph analysis-based logic locking
c432	160	20min	0.03s
c7552	3512	4h30min	0.87s
b19_C	225k	X	1h15min



²Rajendran, Zhang, Rose, Pino, Sinanoglu, Karri *Fault analysis-based logic encryption* IEEE Transactions on Computers, 2013

Security margin

In the final graph, all nodes are available for logic locking.



More nodes can be forced to increase locking strength.

It can make hill-climbing attack and reverse-engineering **harder**.

The designer **tunes** the resources overhead/locking strength **ratio**.

Conclusion

Key points:

- Logic locking is a **powerful** way to make the circuit **unusable**:
 - Very **low** overhead,
 - **Tunable** security margin.
- Graph analysis-based selection method:
 - **Computationally efficient**,
 - Simple **EDA integration**.
- Logic masking/locking alone is **not secure**,
- A **cryptographic primitive** is necessary for security.

Presented at ISVLSI 2015⁴.

All Python scripts are available on the SALWARE project webpage⁵.

⁴ **Colombier, Bossuet, Hély** *Reversible Denial-of-Service by Locking Gates Insertion for IP Cores Design Protection*
IEEE Computer Society Annual Symposium on VLSI, 2015

⁵ <http://www.univ-st-etienne.fr/salware/FOGP.html>

Conclusion

Questions ?

Contact:

b.colombier@univ-st-etienne.fr

