

ANALYSIS OF TRUST-HUB CIRCUITS FOR HARDWARE TROJAN

by

ARATI SHAW

SHREYASI KARAK

SUKANYA NASKAR

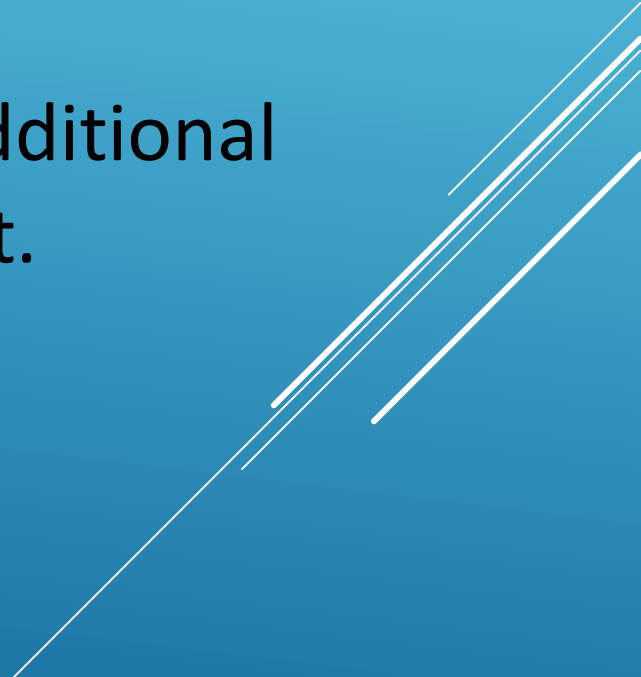
INTRODUCTION

Any hardware circuit requires to perform some predefined functionalities. Malicious circuitry or trojan injected during fabrication or designing may infect the circuit risking the normal functionalities of the hardware.

In addition, such tampering can also lead to leakage of vital information of hardware chip to an unauthorized party.

So, trojan detection is inevitable to counter such malicious modifications of the circuit.

AIM OF THE PROJECT

- ❑ In this project, we are emphasizing on detection of hardware portions where there could be possibility of trojan injection.
 - ❑ Modification of weak links of the circuit using additional hardware so that trojan cannot attack the circuit.
 - ❑ To reduce the vulnerability of integrated circuit.
- 
- A series of three parallel white diagonal lines in the bottom right corner of the slide.

SOME IMPORTANT DEFINITIONS

- ❑ **Trojan:** A hardware trojan is a malicious tampering of certain circuitry of an integrated circuit.
- ❑ **Trust Hub:** It is a circuit used to study the insertion of hardware trojan, it's detection, side channel analysis, vulnerability analysis etc.
- ❑ **Transition Probability:** The probability of occurrence of a transition between two output states of a gate.
- ❑ **Insertion Point:** Point in the circuit which is more susceptible to trojan injection.
- ❑ **Threshold:** A benchmark for transition probability.

OVERVIEW

- ❑ In this project, we will calculate the transition probabilities of each component (gate) of the circuit.
- ❑ This will help us to figure out the low transition probability gates whose output needs to be modified to incur a higher transition probability.
- ❑ The gate(s) connected to most of the low transition probability gates will become the insertion point, where an additional hardware component will be added.
- ❑ Then we will evaluate the modified transition probabilities.

ALGORITHM TO FIND INSERTION POINT(S)

- **INPUT :**

- Circuit Netlist
- List of Low Transition Probability Nodes (L)
- Trigger Threshold (P_{th})

- **OUTPUT :**

- List of Insertion Point (L_p)

ALGORITHM TO FIND INSERTION POINT(S)

Step 1 : For each gate in the circuit with output A(say)

Transition probability of A = signal probability of 0 * signal probability of 1

Step 2 : For each gate(i) with transition probability(TP_i) of the circuit :

if ($TP_i < P_{th}$) : Insert gate(i) in list(L)

Step 3: Convert the circuit into a directed graph where,

node -> gate of the circuit

incoming edges -> input to a gate

outgoing edges -> output from a gate

ALGORITHM TO FIND INSERTION POINT(S)

Step 4: Transpose the graph

Step 5: For each node in L , run depth first search and record the path

Step 6: Determine the common node(k) of all these paths with minimum depth limit and insert node(k) in L_p

D = depth limit of k from each node in L

d = minimum (D)

ALGORITHM TO FIND INSERTION POINT(S)

Step 7: For each node(n) in L : determine node(n_d) at depth d from n

if (atleast one node from n_d is present in L_p)

continue;

else

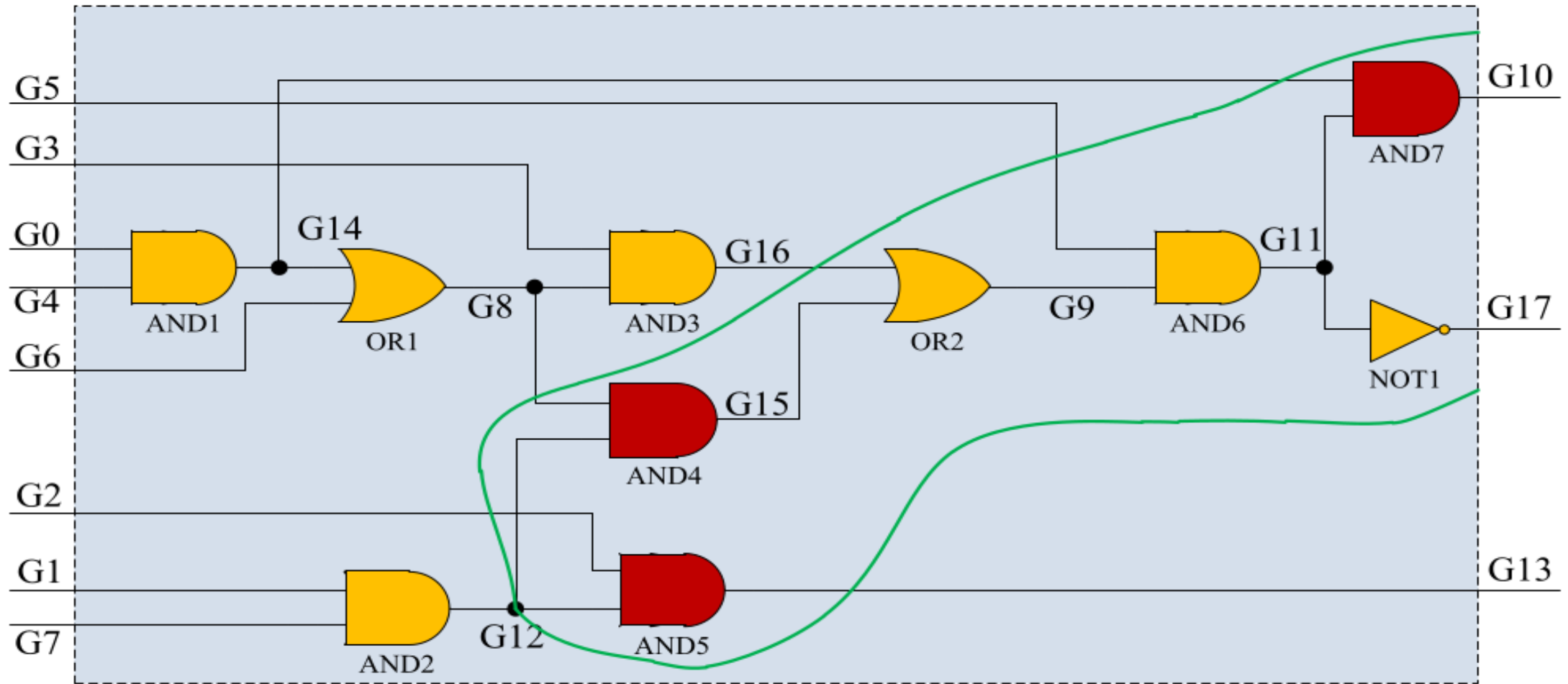
if(signal prob. of 0 of node(n) > signal prob. of 1 of node(n)):

select one node from n_d where ($sp_0 < sp_1$) and insert in L_p

else if(signal prob. of 0 of node(n) < signal prob. of 1 of node(n)):

select one node from n_d where ($sp_0 > sp_1$) and insert in L_p

EXAMPLE CIRCUIT



GATE NUMBER	TRANSITION PROBABILITY (SORTED ORDER)
G10	0.04973500967025757
G13	0.109375
G15	0.1318359375
G11	0.1658773422241211
G17	0.1658773422241211
G14	0.1875
G12	0.1875
G16	0.21484375
G8	0.234375
G9	0.24358749389648438

Threshold Probability : 0.15

Therefore , { **G10** , **G13** , **G15** } are the low transition probability points.

Insertion Points : { **G12** , **G14** }

ADDITIONAL HARDWARE USED

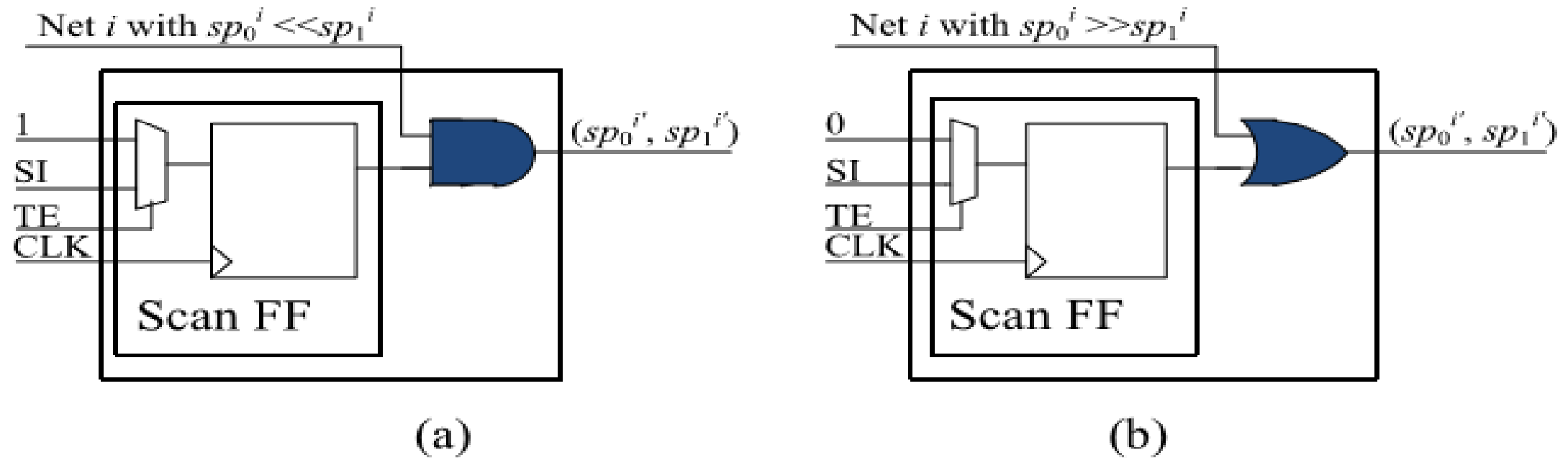


Fig. 1. dSFF structures when (a) $sp_0^i \ll sp_1^i$ and (b) $sp_0^i \gg sp_1^i$.

RESULT

GATE NUMBER	TRANSITION PROBABILITY (sorted order)	Modified TRANSITION PROBABILITY	% change in TP after adding dsFF(s)
G10	0.04973500967025757	0.1875	276.9980165744863
G13	0.109375	0.1875	71.42857142857143
G15	0.1318359375	0.234375	77.77777777777777
G11	0.1658773422241211	0.21185302734375	27.716675769684077
G17	0.1658773422241211	0.25	50.713772386236236
G14	0.1875	0.1875	0.0
G12	0.1875	0.1875	0.0
G16	0.21484375	0.234375	9.090909090909092
G8	0.234375	0.1875	-20.0
G9	0.24358749389648438	0.238037109375	-2.2785999530185577

OUTCOME FROM OTHER BENCHMARK CIRCUITS

BENCHMARK	S15850	S38417	S35932	S38584
OPTIMUM P_{TH}	0.06	0.025	0.07	0.037
NO OF LT NETS	1276	783	2289	1080
NO OF INSERTION POINTS	329	128	512	153
LINS (%)	387	611	447	705
GATE OVERHEAD(%)	6.1	3.78	7.9	4.43

TROJAN ACTIVATION AND DETECTION ANALYSIS BEFORE APPLYING PROPOSED METHOD

(s38417) $P_{th} = 0.025$	TOTAL CIRCUIT ACTIVITY	NO OF TRANSITION ON LT SET	NO OF TRANSITION ON TROJAN INPUTS	NO OF TRANSITION INSIDE TROJAN	NO OF TRANSITION AT TROJAN OUTPUT	TROJAN ACTIVITY	POC	TCA
Trojan 1	486215	206	4	NA	0	0	0	0.0E+00
Trojan 2	486224	214	16	11	0	11	0	2.26E-05
Trojan 3	486238	230	35	25	0	25	0	5.14E-05
Trojan 4	486265	257	89	51	0	51	0	1.04E-04

TROJAN ACTIVATION AND DETECTION ANALYSIS AFTER APPLYING PROPOSED METHOD

(s38417) $P_{th} = 0.025$	TOTAL CIRCUIT ACTIVITY	NO OF TRANSITION ON LT SET	NO OF TRANSITION ON TROJAN INPUTS	NO OF TRANSITION INSIDE TROJAN	NO OF TRANSITION AT TROJAN OUTPUT	TROJAN ACTIVITY	POC	TCA
Trojan 1	215846	713	69	NA	10	10	5	4.63 E-05
Trojan 2	215886	758	143	72	2	74	0	3.42 E-04
Trojan 3	215917	795	184	100	0	100	0	4.63 E-04
Trojan 4	216020	893	357	204	0	204	0	9.44 E-04

CONCLUSION

The proposed methodology is suitable to detect Trojan in hardware circuits. Using the additional hardware (dsFF) we can alter the transition probability of weak links (insertion point) of the circuit and therefore we can resist the malicious attack.

In this way, we can decrease the vulnerability of the integrated circuit chips.

Three parallel white lines of varying lengths are positioned diagonally in the bottom right corner of the slide, pointing towards the top right.

THANK YOU

