Hardware Trojan Detection By Symmetry Breaking In Path Delays

Norimasa Yoshimizu
NanoMason Inc.
Martinez, California 94553

Abstract— **This paper discusses the detection of hardware Trojans (HTs) by their breaking of symmetries within integrated circuits (ICs), as measured by path delays. Typically, path delay or side channel methods rely on comparisons to a golden, or trusted, sample. However, golden standards are affected by inter- and intra-die variations which limit the confidence in such comparisons. Symmetry is a way to detect modifications to an IC with increased confidence by confirming subcircuit consistencies within as it was originally designed. The difference in delays from a given path to a set of symmetric paths will be the same unless an inserted HT breaks symmetry. Symmetry can naturally exist in ICs or be artificially added. We describe methods to find and measure path delays against symmetric paths, as well as the advantages and disadvantages of this method. We discuss results of examples from benchmark circuits demonstrating the detection of hardware Trojans.**

Keywords—hardware trojan; integrated circuits; circuit symmetries; path delay

I. INTRODUCTION

Hardware Trojans (HTs) are malicious changes to integrated circuits (ICs) designed to modify its behavior to the advantage of the adversary, such as to weaken encryption, leak information, or cause failure in designed functions. Modifications might be induced by changing the doping [1] or introducing additional digital and analog circuitry.

The threat of HTs is commonly attributed to the outsourcing of IC manufacturing to foreign states, described at least as early as 2005 [2]. HTs could be damaging to both commercial and national security interests. Research effort into detecting HTs has thereby grown. Side channel methods use secondary physical measurements, such as drive current, to infer the state of ICs and the presence of HTs [3]. Timing methods detect HTs by their effect on path delays in an IC [4, 5]. Logic or functional testing will supply test vectors and determine directly from the outputs whether an HT has modified the IC's functionality from its intended design [6]. Reviews of HTs and their detection are available elsewhere [7-9].

The significant challenge in HT detection is the balance between the sensitivity and speed of the method. ICs are typically very complex and very large, in the sense of numbers of transistors and their wired connections. However, many effective HTs require only minimal modifications to create a very sensitive vulnerability [1, 10]. Finding a handful of malicious transistors in a vast ocean of circuitry is a formidable task. Therefore, techniques must compromise between the test sensitivity and rate.

With these balances in mind, this paper explores a path delay method. Path delay methods will not scale up as well as side channel methods because the former will require partitioned regions. However, this can be counteracted by an increased sensitivity. Commercially available time-based measurements can be far more sensitive (e.g. short-term Allan deviations of 10^{-10} and time resolutions into the tens of picoseconds) whereas electrical current or power measurements will have less precision by up to four to seven orders of magnitude. This higher sensitivity can compensate for a slower testing speed, and may in fact be necessary if the HTs are small. In addition, side-channel measurements use current which is provided throughout an entire IC. Though parts of the IC can be selectively probed by carefully chosen test vectors, there is a constant power consumption that raises the background over which small effects must be detected. Path delays can be measured as purely differences in time, and in fact most timing measurements will improve precision at increasing integration times. And timing methods can target regions of the circuit which are driven by any simultaneously timed signal, such as due to clock-driven flip-flops or nodes with fan-out. These characteristics further increase the sensitivity advantage of time-based methods. Therefore, we believe delay methods to be a powerful tool in HT detection. Any complete HT detection protocol will likely require a combination of methodologies to test a whole IC.

This paper proposes a technique for path delay detection of HTs. First, we are interested in further improving the sensitivity of path delay methods. Path delay methods rely on golden or trusted chips, against which test results are compared. The fabrication process introduces inter-die variations in addition to inter-die variations within the golden chips. The variability in a group of golden chips against the variability in the test chip affects the confidence with which a test chip becomes cleared. Second, we are interested in diminishing the reliance on having a golden chip. The preparation of a golden chip would be very costly because it would be fabricated as part of an untrusted process and require perfect, nondestructive testing to prove that it can be trusted. And rather than a single golden sample, a number of them would be required to calculate the mean and variability of parameters in order to make meaningful comparisons to test chips.

The use of symmetries to detect HTs attempts to mitigate these problems. The symmetry refers to different transistor-level paths that have the same topology. For example, the same subcircuit may have been copied to process the bits on a bus or identical transistor paths may naturally occur. Note the distinction between symmetric subcircuits as compared to symmetric transistor delay path. We are referring to specific transistor level paths and their delays: it is not necessary that the larger logic gate structure be topologically equivalent as well.

The measurement of symmetry by path delay occurs two ways. Suppose logic states $\alpha 1$ and $\beta 1$ have the same delay paths, and separately logic states $\alpha 2$ and $\beta 2$ have the same delay paths. Relative to a clock edge, the delays to the $\alpha 1$ - $\beta 1$ transition edge and to the $\alpha 2$ - $\beta 2$ transition edge will be the same. Or, suppose logic states $\alpha 3$ and $\beta 3$ have the same delay paths and there is a third logic state χ. Relative to a clock edge, the delays to the χ - $\alpha 3$ transition edge and to the χ - $\beta 3$ transition edge will be the same. There are some caveats and cases, particularly of the latter, that are discussed below where the delays may be different. The equivalent delays are possible largely because inter-die variations affect them identically and intra-die variations will be limited if the paths are in close proximity. And note that this does not require a golden chip for comparison, as the comparisons are done within the test chip itself.

The use of self-referencing was explored in [11], which noted that an IC will contain identical regions from a digital logic point of view: for example, an arithmetic logic unit containing equivalent circuitry to separately handle many bits. However, while the same symmetries can be exploited here, there are other symmetries that are not dependent on a signal bus, opening up for wider applicability. Continued work in [12] demonstrates that HTs can be detected as inconsistent power consumption when an IC repeatedly cycles along a closed loop in its finite state diagram but the HT, if using sequential logic, continues to change its state and expend different energies doing so. The authors contend, as we do, that the idea of a golden sample should be abandoned in favor of self-referencing methods in order to greatly increase confidence in the presence of HTs despite growing inter- and intra-die variations throughout smaller technology nodes. However, we believe that time-based measurement methods will be simpler, more sensitive, more flexible, and more targeted than digital / side-channel methods.

This method is limited in that symmetries do not exist everywhere in an IC. Therefore, this cannot be a complete test for HTs. And in addition to the effort required to finding symmetric paths, there is effort required to find the correct test vectors in order to activate and measure desired path delays. Like other path delay methods, scaling up to larger test regions will be a challenge. Therefore, we believe this flexible method serves as a rapid, highly sensitive test for HTs for large swathes of circuitry but should be combined with other HT detection techniques.

This paper discusses the natural existence of symmetries in ICs, methods to find exploitable symmetries, and methods for detecting HTs using these symmetries. We discuss some of our results on benchmark circuits in identifying symmetric paths as well as finding and detecting HTs. We discuss future challenges as well.

II. Delays in Digital Logic

In order to describe delays in complementary metal oxide semiconductor (CMOS) ICs, we first consider the case of a simple NAND gate. In Fig. 1 we show the NAND2 circuit, replacing the PMOS and NMOS transistors with symbols for readability and the dot representing the output node.

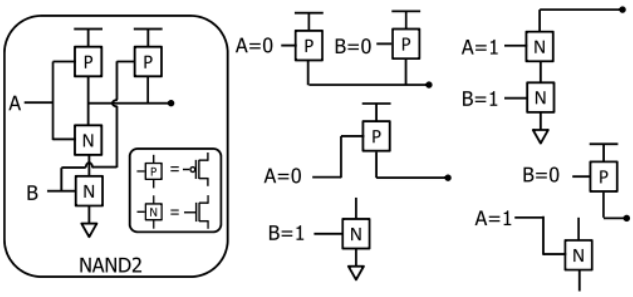

Fig. 1. Two-input NAND gate, showing transistors driving the output denoted by a dot. Note the symmetry in the AB =01 and 10 states.

Note that, like other logic gates, the NAND gate shows symmetry in its AB = 01 and 10 states: a single PMOS driving the output to high. The symmetry between the 01 and10 states is a natural consequence of the commutative property of Boolean algebra. Because the algebra makes no distinction between the two states, the layout looks the same in order to produce the same output. Also note that the A = 1 or B = 1 input is isolated from the output, so that any circuitry behind that input will not affect the path delay. There is non-ideality due to finite off resistance and subthreshold leakage currents. This is a tool that will let us isolate symmetric delay paths from undesired parts of the circuitry so that only the desired paths will be included in the path delay measurement.

III. Illustrative Example

We illustrate the method in a small benchmark circuit: c17 from the ISCAS-85 benchmark circuits [13]. The circuit is shown below, composed of two outputs using six NAND gates from five inputs.

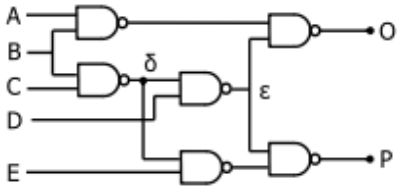

Fig. 2. ISCAS-85 c17 circuit.

At this small scale, symmetries can be found by searching manually through states as in Fig. 1. From the full set of states it is evident that many of them are symmetric, among those that drive the same and different outputs. We choose as an illustration the states ABCDE = {01010, 01100} which produce high and low outputs at O; and the states ABCDE =

{01001, 01000} which produce high and low outputs at P. We label these states as α1, α2, β1, and β2, respectively. Note that there are two symmetries: α1 is symmetric to β1 and α2 is symmetric to β2. Therefore, we expect that the path delay between α1 and α2 and the path delay between β1 and β2 will be the same.

Fig. 3 shows these symmetric states. The insets show the paths along the transistors which are on for each state. These highlighted paths indicate the coverage of the HT detection, in that HTs that are located at these nodes can be detected by this scheme. These four states cover ten of the eleven nodes in the circuit. The insets also help to visualize the gates that are involved and their evident symmetry.

In Fig. 4 we show the clock edge at the input as well as the α and β transitions. For comparison, also shown is another transition (ABCDE = 11010 to 01110) which is symmetric to none of the states in the α and β transitions. These results are from a SPICE simulation using parameters from a 90 nanometer multiple project wafer process run. Variation is shown as multiple transparent lines from ten other runs for each transition. Here and throughout this paper, we added variations of 5% standard deviation in both the width and length of transistors, approximated based on [14-16] and for circuitry that would be close in proximity. As expected, the symmetric transition curves are similar. The small difference is attributed to slight differences in transistors that are switched, for example as discussed in the Section II above. The equivalent path delays would be measured as shown. Also shown are curves for a

Finally, we demonstrate the detection of some HTs. The added HTs were designed to be cause minimal change in logic, changing the output of only one input logic state. Their effects were to change O to O OR D in HT1 and to change the intermediate state at node δ to pass through an OR gate with not D. Each changed a single output state from LOW to HIGH. They are shown in Fig. 5.

In order to detect the HTs, we use the states described above in Fig. 3. The result of the path delays are shown in Fig. 6. The introduction of an HT into the circuit causes the delays α and β to become different, indicating their presence.

Note that the HTs cause a shift in the delay in one of the paths, breaking the symmetry. If the delay paths were extended further while preserving their symmetry, the delays would all simply be increased by a constant amount. However, as the total path length increases, the variation in the path delay also increases, canonically as the square root of the length of the path. Therefore, statistical thresholds will be required to determine if a pair of paths has had its symmetry broken.

We also note that small differences in the transition can be caused by the off-path transistors. For example, the transition due to a pair of series transistors will be faster if one transistor is already on rather than turning both on simultaneously. These differences will affect the precision of the delay measurement and subsequently affect the sensitivity to HTs.

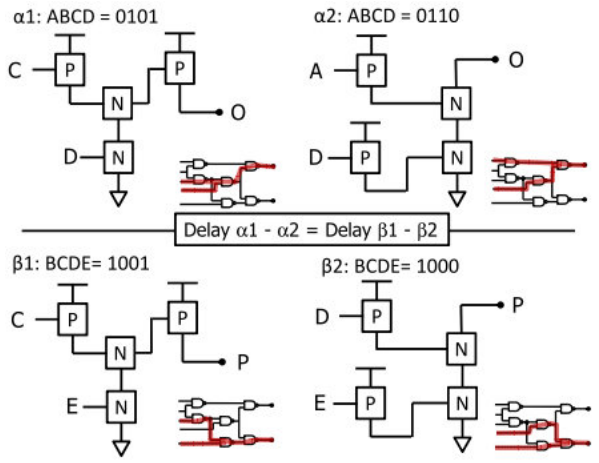

Fig. 3. Symmetric states of c17. The small diagrams next to each state show, as a thick red line, the path of on transistors that determines the net path delay.

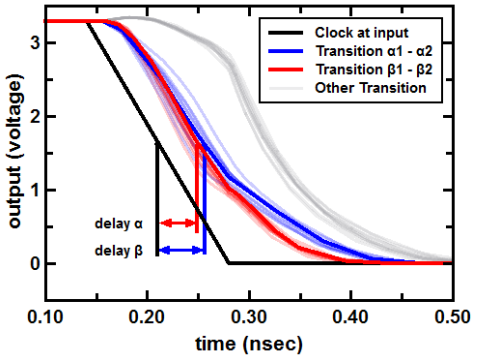

Fig. 4. Transition between states α1 and α2, shown in red, and between states β1 and β2, shown in blue. Due to the symmetry, the transitions delays α and β are the same. Other transition (ABCDE = 11010 to 01110) is shown in grey for comparison. Partially opaque lines demonstrate statistical variations of ten simulations for each transition.

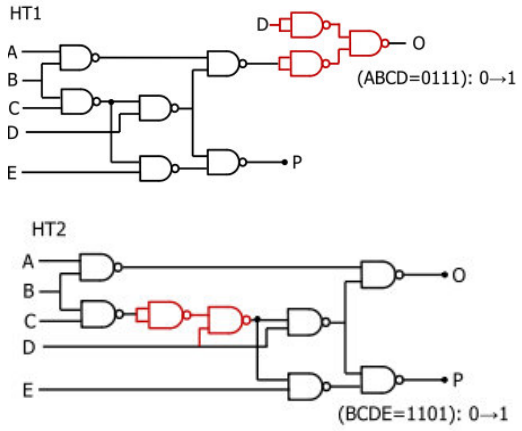

Fig. 5. Hardware Trojans introduced to c17 (Fig. 2) and tested for Hardware Trojans using the transition states described above (Fig. 3). The hardware Trojans are shown in red. The text underneath describes the resulting single change in the output logic, showing the input state and change in output.

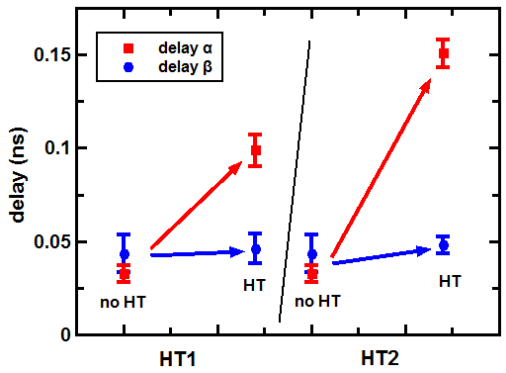

Fig. 6. Detection of Hardware Trojans of Fig. 5 using the symmetric states of Fig. 3. The addition of an HT causes the equal α and β delays to split to different values.

In addition, the selected transition states affect the amount of coverage. Nodes that are used by some but not all the states provide coverage and could indicate the presence of an HT. In contrast if a node were used for all the states, the HT would affect all delay times equally and show no difference. Therefore, selecting transition states that cover the largest number of nodes increases the probability of detecting an HT. For example, the states in Fig. 3 cover ten of the eleven nodes of c17. There are other symmetric states, such as those that connect all four to the same output, which could have been implemented with coverage as few as two nodes. Therefore it is important consider maximizing coverage when choosing the states.

IV. Larger Benchmark Circuit

We also demonstrate the detection of HTs on a larger circuit, c432 from the ISCAS-85 benchmark circuits, which is a 27-channel interrupt controller comprised of 160 gates [13]. We implement its model in an FPGA by using a Xilinx synthesis tool. As in [11], the symmetry existing between bits on a bus can be easily exploited to test for HTs. The output N223 is one of the request pins, and its output is determined by two, 9-bit input buses which undergo the same logic operations.

We test our method by implementing an attempted concealed HT. The output of one of the buses, before a final OR gate, is passed through an AND gate along with an enable input signal. Therefore, if the enable signal is high, the AND gate will pass along the unperturbed output of the bus; if the enable signal is low, the AND gate will force an output of LOW. For these tests, the enable signal is assumed to be high at all times. The results are shown in Fig. 7. The path delays of two different bus bits are used. The splitting in path delays indicates the presence of an HT, though the HT consists only of a single enable AND gate HT that is not triggered.

We also find that symmetry exist between three unrelated inputs: that is, inputs that are not bits on the same bus and do not, overall, undergo the same logic operations to the output. These input bits of three different buses pass through different logic modules to the most significant bit of the channel request output. Nevertheless, they exhibit symmetries in their transistor paths. We inserted an HT similar to the enable AND gate used above in Fig. 7. The results are shown below in Fig. 8.

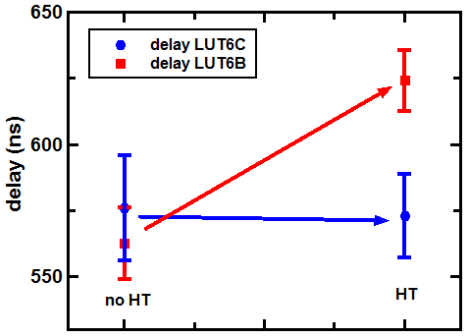

Fig. 7. Detection of single enable AND gate HT in c432. Symmetric paths are of two bits of the buses that undergo the same logic operations to the output N223.

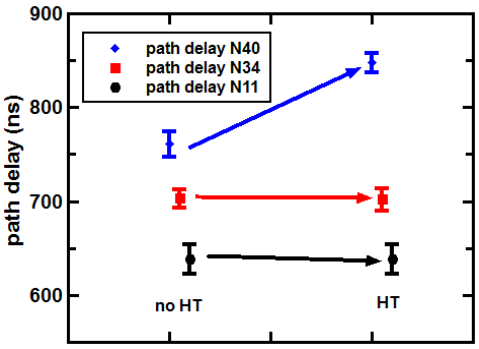

Fig. 8. Detection of a single AND gate HT in c432. Symmetric paths were identified that exist through pins that undergo different logic modules to the channel request bus.

V. Challenges

As in other path-delay methods, the greatest challenges will be in scaling up to larger test ICs.

This method requires the ability to find symmetries in existing circuits or to generate symmetries in circuits during design. This task can be quite complex [17], although it can be greatly simplified in our context. The different logic operations, such as NAND, do not overlap the others because they do not contribute identically to a delay path as they are often designed in CMOS. For example, the OR gate uses two parallel NMOS transistors to ground compared to the AND gate which uses two series NMOS transistors to ground. Therefore, when mixed logic gates are used any two pathways must pass through the same logic gates in the same order in order to result with a symmetric delay path. So, symmetric paths can be found by tracing paths from an output back to the input pins which pass through the same logic gates.

A complete method of laying out all possible states is not feasible when the number of input combinations reach into the thousands or more. Instead, algorithmic methods that start from the output and work back towards the input pins would have to be used to track down symmetric paths. This requires

supporting tools to isolate these symmetric paths when they are discovered. It is also possible that symmetry considerations could be made during the design and layout phase in order to identify them in the early stages, rather than reverse-engineering them as we have had to do here. Symmetries that appear in the layout can be a result of symmetries in multiple-bit logic, apparent symmetries in the logic, or conscious design choices such as to increase the number of logic gates to induce symmetry in parts of the circuit.

However, as we have tried to demonstrate here, unintended symmetries appear to naturally occur in ICs. One limit is to consider a truth table with N input bits that has 2^N possible outputs. There are a maximum of only N+1 outputs, each having between zero and N bits designated to be high, until symmetry will begin to appear. Symmetry will then appear: for example, the output $\overline{A}B \cup \overline{C}D$ would show symmetries in the circuit of two inverters and two AND gates which fan-in to an OR gate. These symmetries evident in the truth tables lead naturally to symmetries in the circuitry. Another example is in Fig. 2, where the output $P = \overline{BCD} \cup \overline{BCE}$ and leads to apparent symmetries.

In addition to finding symmetries, the detection and assertion of symmetry breaking will require statistical considerations. As technologies reach smaller nodes, even their intra-die variations may become overwhelming. As paths get longer, even though the change in path delay caused by an HT stays constant, the variation in even the nominal path delay will increase canonically as the square root of the total length leading to more difficult analyses of symmetry. Furthermore, the off states become less ideal as subthreshold leakage currents and other effects become stronger. These will allow off-path transistors to begin to affect the path delay measured through the desired symmetry paths.

Finally, we note that we have not considered here layout and routing effects and how they may affects the path delay.

VI. CONCLUSION

We have described methods and principles for detecting hardware Trojans by their breaking of symmetries in integrated circuits, as measured by path delays. Symmetries can be found by searching through logic states, or through the logic gate schematics finding equivalent paths through them. In the latter case, the delay paths are isolated by input states (such as a high input node in NAND gates or a low input gate in OR gates). Paths which propagate through different nodes increase the coverage and therefore the probability of detecting an HT. This paper provides some evidence that these desired symmetries can occur not only due to conscious design implementation, but also occur naturally in digital integrated circuits.

These methods can be especially powerful to increase the sensitivity of HT detection, even at some cost in detection speed. In addition to the high sensitivity of time-based measurements, the use of symmetries allows one to abandon the need for golden samples which, especially towards smaller technology nodes, results in lower confidence due to inter-die variability. Tools such as algorithms to find symmetric, high coverage delay paths and test vectors which isolate them can increase the scale of size.

REFERENCES

[1] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy dopant-level hardware Trojans," Proceedings of the 15th Internatinal Conference on Cryptographic Hardware and Embedded Systems (CHES) 2013, pp. 197-214.

[2] Defense Science Board, "High performance microchip supply," 2005.

[3] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," IEEE Symposium on Security and Privacy (SP) 2007, pp. 296-310.

[4] S. Wei and M. Potkonjak, "Malicious circuitry detection using fast timing characterization via test points," Hardware-Oriented Security and Trust (HOST) 2013, pp. 113-118.

[5] Y. Jin, Y. Makris, "Hardware Trojan detection using path delay fingerprint," Hardware-Oriented Security and Trust (HOST) 2008, pp. 51-57.

[6] S. Jha and S. Jha, "Randomization based probabilistic approach to detect trojan circuits," 11th IEEE High Assurance Systems Engineering Symposium (HASE) 2008, pp. 117-124.

[7] M. Beaumont, B. Hopkins, and T. Newby, "Hardware trojans – prevention detection, countermeasures (a literature review)," DSTO-TN-1012, 2011.

[8] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware trojan: threats and emerging solutions," High Level Design Validation and Test Workshop (HLDVT) 2009, pp. 166-171.

[9] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," IEEE Design & Test of Computers, 27 (1) 2010, pp. 10-25.

[10] J. Blömer and J.-P. Seifert, "Fault based cryptanalysis of the advanced encryption standard (AES)," 7th Annual Conference, Financial Cryptography (FC) 2003, pp. 162-181.

[11] D. Du, S. Narasimhan, R. S. Chakraborty, and S. Bhunia, "Self-referencing: a scalable side-channel approach for hardware Trojan detection," Proceedings of the 15th Internatinal Conference on Cryptographic Hardware and Embedded Systems (CHES) 2010, pp. 173-187.

[12] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, "TeSR: a robust temporal self-referencing approach for hardware Trojan detection," Hardware-Oriented Security and Trust (HOST) 2011, pp. 71-74.

[13] M. Hansen, H. Yalcin, J. P. Hayes, "Unveiling the ISCAS-85 benchmarks: a case study in reverse engineering," IEEE Design and Test, 16 (3) 1999, pp. 72-80.

[14] B. Cline, K. Chopra, D. Blaauw, and Y. Cao, "Analysis and modeling of CD variation for statistical static iming," Computer-Aided Design (ICCAD) 2006, pp. 60-66.

[15] I. Ahsan, et. al., "RTA-driven intra-die variations in stage delay, and parametric sensitivites for 65nm technology," VLSI Technology 2006, pp. 170-171.

[16] P. S. Zuchowski, P. A. Habitz, J. D. Hayes, J. H. Oppold, "Process and environmental variation impacts on ASIC timing," Computer Aided Design (ICCAD) 2004, pp. 336-342.

[17] D. Chai, "Circuit Symmetries in Synthesis and Verification," 2009, UC Berkeley, Technical Report No. UCB/EECS-2009-115