Master-slave based test cost reduction method for DNN accelerators

Umair Saeed Solangi$^{1, 2, *}$, Muhammad Ibtesam$^{1, *}$, and Sungju Park$^1, a$)

Abstract To achieve reduction in test time of accelerators, broadcasting of test patterns is used for simultaneous testing of processing elements (PEs). However, number of PEs tested simultaneously is limited because of scan shift power constraint. In this letter, a Master-Slave based test application method is proposed that alleviates this scan shift power constraint. PEs are grouped in Subcores, the tester loads the pattern into Master PE of Subcores. From Master, test patterns are loaded into adjacent Slave PEs of Subcore. By limiting scan shift power to one Master PE per Subcore, more PEs are allowed to be tested simultaneously.

Keywords: artificial intelligence (AI) accelerators, design for testability, fault localization, scan test, testability

Classification: Integrated circuits

1. Introduction

Deep neural network (DNN) accelerators are being developed and implemented with a variety of applications, such as for Cloudcenters/Automotives. The architecture of these accelerators is optimized to suit their application [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17]. A common characteristic of these accelerators is that it consists of an array of identical processing elements (PEs). Where each PE performs multiply-and-accumulate (MAC) operations over activation and weight inputs to generate a sum response. A PE has a combinational logic to perform MAC operation and sequential logic which stores/propagates the sum values. An accelerator can have an array of 100s–10000s of such interconnected PEs. It has been studied that manufacturing defects (stuck-at faults) can reduce the classification accuracy from 74.13% to 39.69% with only 0.005% faulty PEs [18]. A recent study has also demonstrated that classification accuracy drops from 97.4% to 7.75% with a fault rate of 0.0003% in the accelerator [19]. Testing of such a large number of PEs can present some challenges in terms of ATPG effort, test time and test power. Recently, a C-testing approach based on full scan DFT is proposed in [20, 21]. It utilizes the interconnectivity between neighboring PEs for application of test patterns. Multiple test iterations are used in checkerboard style to deliver the test patterns to each PE. This reduces the ATPG effort to a single PE. Logic BIST based STUMPS solution can also reduce the ATE storage overhead, but it suffers from high shift power and aliasing error issues [22]. Industrial test solutions utilize homogeneity of arrays to limit ATPG effort for testing of such large arrays [23, 24, 25, 26, 27, 28]. These solutions use broadcasting of same input test patterns to identical cores. This improves the test time by enabling simultaneous testing of identical cores. However, due to serial scan shift power constraint, the number of cores tested simultaneously is limited. This necessitates for testing the whole array into multiple non-overlapping test sessions and may increase overall test time.

In this brief, a Master-Slave test pattern application method is proposed. It extends simultaneous testing limit, which is otherwise bounded by scan shift power constraint. Key contributions of this letter are as follows.

A novel Master-Slave based test method for homogenous array of DNN accelerators. A clock control unit to enable pattern loading into scan-connected PEs is presented. A sequential unloading mechanism for test responses with simultaneous response checking is proposed that enables fault localization on PE-level. Test time improvement is presented in conjunction with power analysis.

The rest of this letter is organized as follows. Section 2 presents the proposed test pattern application method with modified clock control circuitry. Section 3 presents method for unloading test response along with response checking and fault localization. Section 4 presents the experimental work and results with proposed method. Section 5 concludes the letter.

2. Test pattern application with proposed method

In proposed method, homogeneity of PEs is exploited to limit ATPG effort to a single PE. For application of identical test patterns, PEs are grouped into multiple PEs called Subcore. Each Subcore has one Master PE and multiple Slave PEs. Subcore based partitions for an NxN array is shown in Fig. 1 with 3 Slaves/Subcore. Scan FFs of Master PEs are connected in scan chains. Master is a PE which is connected to tester for shift-in of scan patterns into scan chains. Whereas scan FFs of Slaves are not stitched into chains to allow parallel pattern loading. For this, parallel connections among respective scan FFs are made, this connection is called scan-connection. Slaves are scan-connected with Master and neighboring Slave. In Fig. 1, 2 PEs, i.e., 1 horizontally adjacent (Slave-1) and 1 vertically adjacent (Slave-2) with Master PE are connected, while remaining Slave PE (Slave-3) is connected to horizontally adjacent Slave (Slave-2). This limits the fanout for scan-connection to 2 for any Subcore size. In an accelerator array, the sum response is propagated vertically through columns of PEs.

$^1$ Dept. of Computer Science & Engineering, Hanyang University, Seoul, 04763, South Korea
$^2$ Dept. of Electronic Engineering, Quaid-e-Awam University of Engineering Science & Technology, Larkana 77150, Pakistan
$^*$ Contributed equally.
$a$) paksj@hanyang.ac.kr

DOI: 10.1587/elex.18.20210425
Received October 9, 2021
Accepted November 10, 2021
Publicized November 22, 2021
Copyedited December 25, 2021

Copyright © 2021 The Institute of Electronics, Information and Communication Engineers
Due to this reason, the scan-connection among Slaves is done horizontally to avoid routing congestion among PEs. Generally, parallel pattern loading methods use common source/sink of test patterns/response for the device under test (DUT) [29, 30]. This increases interconnect length, routing complexity and routing area overheads. In proposed method, Masters are sources of test patterns in each Subcore and routing (scan-connection) is distributed homogenously throughout the array. The length of scan-connections is also equal to PE-PE, which is same as interconnect length of activation/weight/sum. Detailed scan-connection in a Subcore is shown in Fig. 2, where each number of scan FF relates to the serial order of scan chain FFs of Master PE. Scan-ins of Slave scan FFs are connected to output of each scan FF of Master/Slave.

Since Master has scan chains, multiple test clock cycles are required to load the test pattern into the Master with lower scan shift frequency (to limit the scan shift power). Whereas a single test clock cycle loads the test pattern (in parallel) into Slaves, this cycle is called load-cycle. Each Slave is loaded with same test pattern with subsequent load-cycles. So, no scan shift is required for pattern loading into Slaves, which results in zero scan shift power for the Slaves. Hence, scan shift power is limited to only Master PE in a Subcore. There will be lesser switching power with parallel pattern loading into Slaves.

Fig. 3 shows clock control unit for Subcores. An example Subcore with 3 Slaves is shown. Two counters are used, 1) Scan counter and 2) Slave counter. When ‘Test Mode=1’, scan counter runs for the length of scan chain of Master. When test pattern from tester is completely shifted into scan chain then pulse shift enable ‘PSE’ signal becomes ‘1’. This inhibits clock propagation to Master. ‘PSE=1’ starts Slave counter, which runs for the length of number of Slaves (3 in this case). This counter generates control signals ‘S0S1’ to control the clock propagation to Slaves in phase shifted manner. Fig. 4 shows phase shifted clock pulses for each Slave when Slave counter goes through ’S0S1=00→01→10’ states. From tester unit, additional test cycles are required as load-cycles for Slaves, where number of load-cycles is equal to number of Slaves of Subcore, as shown in Fig. 5. This clock control incurs a performance overhead of an ‘AND’ gate over the clock tree. In practice, tester frequency can go up to 1 GHz, which is usually lowered to control scan shift power. Since Slaves load the test patterns in parallel, higher tester frequency can be utilized for load-cycles for Slaves. This will result in further test time reduction.
3. Sequential unloading of test responses with fault localization

In the proposed method the testing is done in 2 phases, 1) Master response is tested by tester and then 2) Slaves are tested by logic comparison against Master’s response in a Subcore as shown in Fig. 6. For unloading test responses, Master PE shifts out test response while simultaneously loading next test pattern via scan chains. As Slaves have no scan chains, extra logic is used to unload the test response. A scanout multiplexer FFs. A single scan chain counter is used (per array) and a single response comparison unit (per Subcore). Control pins of ‘MUX’ are driven by scan chain counter. For unloading the test responses, when ‘Test Mode=1’ and ‘PSE=0’, the scan chain counter runs (for the length of scan chain). This enables tester clock propagation to scan chain counter. When ‘PSE=1’, the sequential unloading of test response from Slave is stopped. This is to synchronize the serial loading/unloading of test pattern into Master with sequential unloading from Slaves. Each counter state will scanout the scan FFs response from each PE (Master/Slaves) via ‘MUX’. MUX logic is tested along with scan chain integrity test by ‘001100’ pattern. It is the first pattern to be loaded into Subcores, passing this test results in validation of scan chains of Masters and MUX logic connected with Slaves. This is stopped. This is to synchronize the serial loading/unloading of test pattern into Master with sequential unloading from Slaves. Each counter state will scanout the scan FFs response from each PE (Master/Slaves) via ‘MUX’. MUX logic is tested along with scan chain integrity test by ‘001100’ pattern. It is the first pattern to be loaded into Subcores, passing this test results in validation of scan chains of Masters and MUX logic connected with Slaves.

3.1 Fault localization with proposed method

The response comparison is done bit-by-bit among PEs (of Subcore) test pattern responses. Per clock comparison increases fault detection rate [22]. The output of response comparison unit is Test Pass/Fail flag (shown in Fig. 7), this response compression reduces ATE storage [31]. Flag is ‘1’ in case of mismatch among response bits of PEs and detects faults in a Subcore. This results in fault localization on Subcore-level. To localize the fault on PE-level, the test pattern which results in fault is applied again and load-cycle is restricted to 1 cycle (loads test pattern in Slave-1). This allows checking of Slave-1 against Master. If there is no mismatch during response unloading, then Slave-1 is classified as non-faulty. Then same test pattern is loaded with 2 load-cycles to include Slave-2. This sequence is repeated to include all Slaves with subsequent load-cycles and response comparison is done until Slave-n, which correlates with the number of n-th load-cycle used, thereby locating the faulty PE. This PE-level fault localization can improve the yield by enabling pruning and skipping of faulty PEs during DNN operation [20].

4. Experimental results and evaluation

Experimental evaluation of proposed method is performed for Tesla’s NPU [1] and Google’s TPU [2, 3, 4, 5, 6]. The main difference is that the former uses systolic array, and the latter uses non-systolic array. Both have 8-bit activation and weight inputs per PE. Synopsys DFT Compiler is used to implement full scan DFT. Scan chain length of 16 (per Master PE) is inserted for TPU and NPU with 2 and 3 scan chains, respectively. With Synopsys Tetramax ATPG, a single PE for TPU requires 34 patterns with 99.83% coverage. Whereas PE of NPU requires 39 patterns with 99.96% coverage. It is shown in Fig. 8 & Fig. 9, the parallel loading for test patterns result in much less switching power in Slaves compared to serial loading of test patterns into Master. The underlying reason is that the switching power correlates with
switching transitions, which is proportional to multiple shift cycles for Master. For parallel loading in Slaves, switching transitions are restricted to single load-cycle. In the proposed method, this power difference is leveraged by combining multiple Slaves with Master into Subcores for test pattern loading.

Proposed method is compared with Broadcast method [23, 24, 25, 26, 27, 28]. In Broadcast, group of PEs are applied with test patterns (parallel) in multiple test sessions. For proposed method, number of Subcores for a given array with specified power constraints (switching and peak) is determined with Algorithm 1. Synopsys Primepower is used for power analysis with VCD files of pattern simulations. For NPU, P_M = 6.14E-07 W and P_S = 2.49E-07 W with K = 2.47, this allows 3-Slave/Subcore. Master peak power = 8.84E-03 W and P_KS = 8.46E-03 W. So, the ratio is ≈ 1 which implies that if ‘n’ number of PEs are tested per session (in Broadcast) then ‘1xn’ number of Subcores can be tested per session. For TPU, P_M = 1.11E-06 W and P_S = 1.75E-07 W with K = 6.3, this allows 7-Slave/Subcore. Master peak power = 6.87E-03 W and P_KS = 8.14E-03 W. So, the ratio is ≈ 0.8 which implies that if ‘n’ PEs are tested per session (in Broadcast) then ‘0.8xn’ Subcores can be tested per session.

A 3-Slave Subcore will require 3 load-cycles/pattern and a 7-Slave Subcore will require 7 load-cycles/pattern in addition to 16 scan shift cycles. Scan shift power limit is set by limiting the number of PEs tested simultaneously to 1) fixed number of PEs i.e., 8 and 16 (these limits are conservative) and 2) relative to array size i.e., 50%. In 2nd case the proposed method with 3-Slave method will allow 3x DUT/test sessions, while 7-Slave evident that proposed method yields test time improvement irrespective of array size. Test time improvement is extended by utilizing higher tester frequency for load-cycles i.e., 500 MHz.

### Algorithm 1 for Determining no. of Subcores tested/session

1. **Input** total no. of cores 'N'
2. **Input** peak power limit of Array core = 'P_M'
3. **Input** peak power of Slave core = 'P_S'
4. **Input** switching power for a single Master PE for test pattern = 'P_s'
5. **Input** switching power for Slave unit = 'P_s
6. **Calculate** \( T_{slave} = K(S_{slave} \text{ units per Subcore}) \)
7. **Roundoff** K to a whole number
8. **'N'** (K+1)= Ns (no. of subcore)
9. **For** (n = 1 to Ns <= P_M <= P_s)
10. Subcore tested per session = Ns/n

### Table I

| Array Size | PE limit/ test session | TPU | Test Cycle Method | 8 | 16 | 50% |
|------------|------------------------|-----|-------------------|---|----|-----|
| 8x8        | 4752                   | 16  | Broadcast         | 10 MHz | 500 MHz | 10 MHz | 500 MHz |
| 16x16      | 19080                 | 748 | 2576              | 520     | 374225   | 1188   | 832    |
| 52x32      | 76032                 | 11975 | 38016           | 8320    | 5987.6   | 1188   | 832    |
| 64x64      | 304128                | 47900 | 152064          | 33280   | 23950.4  | 1188   | 832    |
| 128x128    | 1216332               | 59680 | 266240          | 133120  | 95801.6  | 1188   | 832    |
| 256x256    | 4866048               | 766412 | 2433024        | 532480  | 383206.4 | 1188   | 832    |

| Test Time Improvement | 78.1% | 83.4% | --  | 78.1% | 83.4% | --  |

### Table II

| Array Size | PE limit/ test session | NPU | Test Cycle Method | 8 | 16 | 50% |
|------------|------------------------|-----|-------------------|---|----|-----|
| 8x8        | 5432                   | 1358 | Broadcast         | 10 MHz | 500 MHz | 10 MHz | 500 MHz |
| 16x16      | 21728                 | 1362 | 2716              | 796     | 681     | 1358   | 679    |
| 52x32      | 890912                | 5450 | 38684             | 10864   | 27235   | 1358   | 679    |
| 64x64      | 347648                | 104888 | 182211         | 5384    | 27235   | 1358   | 679    |
| 128x128    | 13905592              | 407552 | 388846        | 695296  | 205776  | 1358   | 679    |
| 256x256    | 5526368              | 163020 | 1395384       | 2781184 | 81504   | 697692 | 1358   | 679    |

| Test Time Improvement | 70.6% | 74.9% | --  | 70.6% | 74.9% | --  |

4.1 Area overheads

The ‘MUX’ overhead is 8.97% for TPU and 9.45% for NPU. Scan chain counter for an 8x8 array, has 0.03% area overhead for TPU and 0.02% for NPU. Response comparison logic for a 3-Slave incurs an overhead of 0.19% for TPU and 0.10% for NPU. The clock control unit for an 8x8 array has an overhead of 0.02% for TPU and 0.01% for NPU. Compared to serial
full scan the proposed method incurs only 1 extra fanout/scan FF. And since scan-connection is made among neighboring elements, the addition of scan interconnects adds relatively less routing area in already heavily interconnected array. The interconnect area overhead is 4.7% for TPU and 2.12% for NPU frequency while not exceeding the given shift power limit (which are set for parallel test method).

5. Conclusion

In this letter, a novel test methodology for testing an array of PEs was proposed to alleviate shift power limits that extends the limitation of simultaneous testing. Test time improvement is proportional to Subcore size and is independent of array size. PE-level fault localization also aids in yield improvement.

Acknowledgments

This work was supported in part by the Higher Education Commission, Govt. of Pakistan, under the scholarship program titled “Faculty Development of UESTPs/UETS”. This research was also supported by the BK21 FOUR (Fostering Outstanding Universities for Research) funded by the Ministry of Education (MOE, Korea) and National Research Foundation of Korea (NRF).

References

[1] P.J. Bannon: U.S. Patent 0 026 078 A1 (2017).
[2] N.P. Jouppi, et al.: “In-datacenter performance analysis of a tensor processing unit,” 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA) (2017) 1 (DOI: 10.1145/3079856.3080246).
[3] J. Ross, et al.: U.S. Patent 9747546 B2 (2015).
[4] J. Ross and A. Phelps: U.S. Patent 9697463 B2 (2015).
[5] J. Ross: U.S. Patent 9805304 B2 (2015).
[6] J. Ross and G. Thorson: U.S. Patent 9747548 B2 (2015).
[7] V. Sze, et al.: “Efficient processing of deep neural networks: A tutorial and survey,” Proc. IEEE 105 (2017) 2295 (DOI: 10.1109/JPROC.2017.2767140).
[8] J. Chen and X. Ran.: “Deep learning with edge computing: A review,” Proc. IEEE 107 (2019) 1655 (DOI: 10.1109/JPROC.2019.2921977).
[9] NVIDIA: “JETSON TX2 high performance AI at the edge,” (2019) https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/.
[10] Google: “Edge TPU run inference at edge,” (2019) https://cloud.google.com/edgetpu/.
[11] F. Wang, et al.: “Deep learning for edge computing applications: A state-of-the-art survey,” IEEE Access 8 (2020) 58322 (DOI: 10.1109/ACCESS.2020.2982411).
[12] J. Chen and X. Ran.: “Deep learning with edge computing: A review,” Proc. IEEE 107 (2019) 1655 (DOI: 10.1109/JPROC.2019.2921977).
[13] M. Sankaradas, et al.: “A massively parallel coprocessor for convolutional neural networks,” Proc. 20th IEEE Int. Conf. Appl.-Specific Syst., Archit. Processors (2009) 53 (DOI: 10.1109/ASAP.2009.25).
[14] S. Chakrabhuti, et al.: “A dynamically configurable coprocessor for convolutional neural networks,” Proc. 37th Annu. Int. Symp. Comput. Archit. (ISCA) (2010) 247 (DOI: 10.1145/1815961.1815993).
[15] Y. Chen, et al.: “Eyeries v2: a flexible accelerator for emerging deep neural networks on mobile devices,” IEEE J. Emerg. Sel. Topics Circuits Syst. 9 (2019) 292 (DOI: 10.1109/JETCAS.2019.2910232).
[16] T. Chen, et al.: “DianNao: a small-footprint high-throughput accelerator for ubiquitous machine-learning,” SIGARCH Comput. Archit. News 42 (2014) 269 (DOI: 10.1145/2565482.2541967).
[17] L. Cavigelli, et al.: “Origami: a convolutional network accelerator,” Proc. 25th Ed. Great Lakes Symp. VLSI (2015) 199 (DOI: 10.1145/2742060.2743766).
[18] J.J. Zhang, et al.: “Analyzing and mitigating the impact of permanent faults on a systolic array based neural network accelerator,” Proc. IEEE 36th VLSI Test Symp. (VTS) (2018) 1 (DOI: 10.1109/ VTS.2018.8368656).
[19] S. Kundu, et al.: “Toward functional safety of systolic array-based deep learning hardware accelerators,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 29 (2021) 485 (DOI: 10.1109/TVLSI.2020.3048829).
[20] A. Chaudhuri, et al.: “C-testing and efficient fault localization for AI accelerators,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. (2021) (DOI: 10.1109/TCAD.2021.3107401).
[21] A. Chaudhuri, et al.: “C-testing of AI accelerators,” Proc. IEEE 29th Asian Test Symp. (ATS) (2020) 1 (DOI: 10.1109/ATS49688.2020.9301581).
[22] Y. Yu, et al.: “Efficient concurrent BIST with comparator-based response analyzer,” Proc. IEEE Int. Instrum. Meas. Technol. Conf. (I2MTC) (2013) 1115 (DOI: 10.1109/I2MTC.2013.6555587).