Enhanced Embedded Test Compression Technique For Processing Incompressible Test Patterns

Sebastian Huhn*† Stephan Eggersglüß‡ Rolf Drechsler*†

*University of Bremen, Germany
{huhn,drechsle}@informatik.uni-bremen.de

†Cyber-Physical Systems, DFKI GmbH
28359 Bremen, Germany

‡Mentor Graphics - A Siemens Business
21079 Hamburg, Germany

Abstract—**Today, complex circuits are used for various safety-critical applications which require zero defects during manufacturing. This policy implies that numerous test patterns have to be executed to achieve the required test coverage. This leads to a high test data volume and, thus, to increased test costs. Embedded test compression techniques are introduced into the circuit in order to mitigate this effect. The major share of the test patterns can be significantly compressed. However, depending on the test application (e.g. low pin count test), parts of the tests are incompressible due to the architecture and will be rejected. This leads to a test coverage decrease which, in turn, jeopardizes the zero defect policy. Therefore, the rejected test patterns are typically transferred in an uncompressed way while bypassing the compression architecture. However, this is extremely costly. This work proposes a novel architecture that seamlessly combines the advantages of an embedded test compression technique with a lightweight codeword-based compression scheme, which is meant to process specifically the rejected test patterns. The proposed architecture mitigates the adverse impact of rejected test patterns on the compression ratio of state-of-the-art techniques. First experiments already show that a significant compression ratio up to 87.48% is achievable.**

I. Introduction

The latest progress in the field of design and manufacture of *Integrated Circuits* (ICs) enables completely new fields of application. For instance, the newest generation of *Electronic Control Units* (ECUs) integrates a huge number of highly complex ICs to implement advanced driver-assistance systems. Due to the safety-critical aspect of these applications, potential defects have to be excluded, which could have been occurred during the manufacturing. Thus, a very high test coverage is required leading to large sets of test patterns, which have to be executed during the manufacturing test.

Complex designs utilize powerful embedded test compression techniques [1], [2] to cope with this high volume of test data aiming to reduce the required testing time and, finally, to save costs. Such a technique achieves a significant compression ratio by, e.g., taking advantage of certain structural properties of the test data. This works fine for the majority of the test patterns. Even though several enhancements (like [3]) have been proposed in the past, one shortcoming remains: A certain amount of the test patterns cannot be compressed at all; they are rejected by the compression architecture and, thus, are not compressed. Due to the high requirement of the test coverage, the *Rejected Test Patterns* (RTP) are crucial as well and cannot be simply neglected. Thus, they have to be applied during the test to avoid any loss of test coverage. Typically, these RTPs are then transferred to the chip in a sequential and completely uncompressed fashion, which yields to an adverse impact on the overall compression ratio as well as to a significant test cost increase.

This work proposes a novel architecture that seamlessly combines a state-of-the-art embedded test compression technique with a codeword-based compression scheme. The codeword-based approach is meant to be applied for the RTPs to tackle the shortcomings of existing techniques. More precisely, the adverse impact of RTPs on the overall compression ratio is significantly reduced. Furthermore, the introduced hardware overhead is negligible and a powerful retargeting approach has been implemented on basis of state-of-the-art formal techniques as proposed in work [4]. First experiments already show that a compression ratio up to 87.48% for the RTPs can be achieved.

II. Preliminaries

Within the last decade, different test compression techniques have been proposed in the literature [1]–[3], which all aim to reduce the test data volume and the test application time, respectively. To take advantage of these methods, it is required to embed dedicated hardware on-chip to enable the transfer of *Compressed Test Patterns* (CTPs) from the (external) test equipment to the circuit-under-test. More precisely, the newly inserted hardware implements, among others, an on-the-fly decompressor $\mathcal{D}$, which allows decompressing the CTPs on-chip without any loss of information. Typically, the tests are determined by automatic test pattern generation tools. These patterns are then post-processed by a *retargeting* procedure applying the actual compression scheme on the original test patterns resulting in the final CTPs.

Commercially available state-of-the-art compression techniques achieve a significant compression ratio by taking advantage of certain structural properties of the test data. The compression ratio of these techniques scales with the share of *unspecified-values* (X-values) and, hence, either a large number of X-values has to be included in the test data or the generation of test cubes has to be considered in the compression procedure [3] forming a run-time intensive task. This works fine for many of the test patterns, however, the remainder of the test patterns cannot be compressed at all; they are rejected due to the introduced compression architecture. It is not possible that they are decompressed by $\mathcal{D}$ on-chip without any loss of information, which is due to certain structural properties of these patterns. These RTPs have then to be transferred via a dedicated Bypass Module $\mathcal{B}$ in an uncompressed and sequential fashion, which bypasses the $\mathcal{D}$-infrastructure. Even if this uncompressed transfer results in a significant overhead in test time as well as test data, this is crucial to avoid any loss in test coverage.

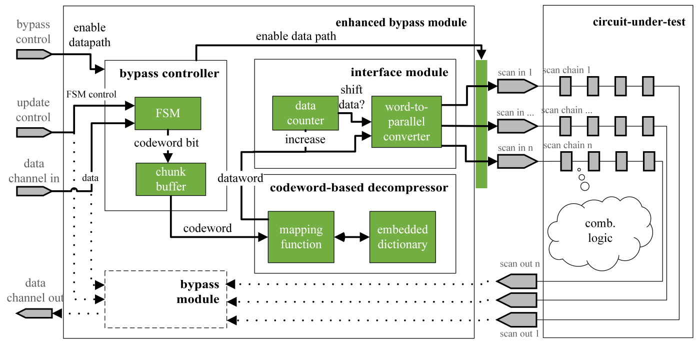

Figure 1: Compression Architecture

Other powerful compression techniques, which are meant to be used for fully-specified, high-entropic data, invoke statistical [5] or even codeword-based compression schemes [6]–[9]. This means that an embedded dictionary is utilized, which contains multiple, individually configurable codewords. More precisely, the dictionary realizes a mapping function ψ such that $\psi(c_i) \rightarrow u_i$ holds, i.e., every single (short) codeword c_i is projected to a (long) dataword u_i. Analogously to other techniques, the original test data have to be post-processed by a retargeting tool, which emits the compressed data, i.e., a sequence of codewords $c_1 ... c_n$. This sequence can be restored by ψ to the original test data without any loss of information on-chip, which, of course, requires an implementation of ψ as a part of the codeword-based decompressor in hardware.

III. Enhanced Compression Scheme

The proposed approach is briefly described in this section and shown in Figure 1. The basic idea of the proposed approach is about introducing a codeword-based decompressor instead of a simple bypass module as generally applied to address RTPs. The proposed scheme is shown in Figure 1 and focuses on the incoming-data of RTPs, which have to be retargeted once prior to the test application. The retargeted patterns, i.e., a sequence of codewords, are transferred bit-wise to the circuit. When a codeword is completed, the decompressor expands it to the (original) dataword and temporarily stores it until enough data are available to feed every input of the scan chains simultaneously. This is a significant improvement over the regular bypass, since it can feed the test data into the scan chains in parallel and not serially as the regular bypass does.

As stated, the three main modules are required to realize the prosed enhanced compression scheme, which are described in the remainder of this section.

1) The *bypass controller* contains a *Finite-State-Maschine* (FSM) as briefly shown in Figure 2, whose state transitions are controlled by the external update_control signal. The FSM design allows differentiating between data and instructions (*inst*). For instance, an instruction is used to (de-)activate the enhanced compression scheme, i.e., the regular bypass is still operational[1]. The current implementation covers four different instructions and, consequently, these instructions are encoded by opcodes holding two bits. However, this length can be easily extended to support further instructions if required. Certain states exist to process either a single bit of a codeword (chunk) or a completely received codeword. Latter state allows *pausing* the transfer if required. This separation allows utilizing codewords concurrently that are not free of prefixes that increases the encoding effectiveness. The chunks are stored in the *chunk buffer* until the codeword transfer has finished and the complete codeword is available. A further stage of *launch* states for the data or instruction yields to an easy control of the dataflow.

2) The *codeword-based decompressor* implements the embedded dictionary, which holds the codewords c_i in conjunction with the associated (uncompressed) dataword u_i. The capacity of different codewords that can be stored within this embedded dictionary depends on the maximal codeword length. Increasing this length leads to a higher capacity. However, in turn, this leads also to increased hardware costs since the embedded dictionary has to be implemented on-chip. A maximal codeword length of three bits is assumed in this paper since this bound has been identified as a good trade-off between compression effectiveness and hardware costs [7]. Consequently, the embedded dictionary holds $\sum_{i=1}^{3} 2^i = 14$ entries, which can be dynamically configured. Every entry consists of one binary-encoded, unique codeword c_i (with a length of 1 to 3 bit) and an associated (binary) dataword u_i. This dataword u_i tends to have a greater length $|u_i|$ since this enables the data volume reduction, i.e., the compression. To keep the resulting hardware costs low, a maximum dataword length of 8 bit is assumed. Furthermore, this module implements the mapping function $\psi(c_i) \rightarrow u_i$, which provides the functionality for the later decompression.

3) The *interface module* realizes the junction between the newly introduced codeword-based decompressor and the inputs of the scan-chains. In this scenario, the codeword-based decompressor acts as the data source by decompressing newly received codewords to the associated datawords using Ψ and the inputs of the scan-chains act as the data sink. This module is important to feed the scan data into the scan chains in parallel differently to the regular bypass, which works in a serial manner. As stated, the codewords of the embedded dictionary can be configured to arbitrary datawords (within the assumed boundary). Consequently, the length of the individual dataword is not necessarily the same, which ensures a high flexibility and, thus, achieves high compression effectiveness. Due to this fact, a mechanism is required that keeps track of the actual number of available data bits. This information is then used to decide whether enough data bits are available to execute a parallel shift to all inputs of the scan-chains. If enough data bits are available, the shift operation is performed or, otherwise, more data bits are received until the required amount of data is available.

Three different components are required to implement the interface module as follows: A data counter keeping track of the currently decompressed datawords, i.e., the number of available bits at this point of time, a capture register storing the currently decompressed bits and an update register holding the bits that are shifted into the scan-chains. After all chunks of a codeword c_i have been received, c_i is decompressed into the (uncompressed) dataword u_i and stored in the capture register. The counter is then increased by the length of the newly decompressed dataword, i.e., $|u_i|$. If the counter exceeds the required value of N, the parallel shift operation is performed by transferring the N least recently received bits from the capture

[1]Note that the regular wiring is not completely shown in Figure 1.

register to the update register, which is then used as the data source. The counter is then decreased by N.

IV. Experimental Evaluation

This section describes the experimental evaluation of the proposed technique, which has been done using the *netcard* circuit from the *IWLS 2005* benchmark set [10], which holds 97.831 (96.569) sequential (scannable) elements.

The underlying scan capabilities as well as the embedded test compression have been both introduced by a commercial tool[2]. As indicated earlier, the number of RTPs might be significant [11]. However, it depends on the test application and also may differ between different synthesis runs, e.g., applications like the low pin count test holds typically a very high ratio of RTPs. In order to have reproducible results, we therefore consider in the following all test patterns for the experimental evaluation. The hardware costs of the proposed enhanced compression architecture are within the same order as the regular embedded compression hardware, which has been validated by a commercial synthesis tool.

The proposed codeword-based compression module is solely implemented in Verilog and the developed module is then instantiated within $\mathcal{B}$, whereby various Verilog *param* allow an easy adoption to further benchmarks. The test patterns, which are generated by a commercial tool, are then processed by the retargeting framework of work [12], which has been slightly adopted. Among others, it is combined with a partitioning approach of work [4] to keep the run-time manageable while further improving the effectiveness. Hence, data blocks with a size of 4k are considered for the retargeting. The retargeting is executed on an *Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz* with *32 GB* system memory within a *C++* compiler-environment (*gcc-Version 8.2.1*). Finally, a commercial tool has been used to validate the (simulated) transfer and decompression on-chip by the newly introduced enhanced compression module.

Note that the configuration of the embedded dictionary is applied once prior to the data block transfer. The configuration bits are included in the shown benchmark results. Furthermore, the proposed enhanced compression module is complete by construction, i.e., any arbitrary RTP can be transferred and, thus, no test coverage loss is introduced.

The average retargeting time per test pattern requires 37.8 seconds. Since the retargeting has just applied once after the test pattern generation process, the run-time is manageable and, furthermore, the retargeting invokes currently only one single thread and the individual blocks are objects to be parallelized. The proposed enhanced embedded test compression technique allows an average (min./max.) compression ratio of 43.97% (29.81% / 87.48%) per test pattern. Besides this, the achieved compression ratio is stable over the conducted experiments ($N = 9899$), which is indicated by a variance σ of 0.006 regarding the compression percentage.

V. Conclusions

This paper presents a novel architecture for test compression, which allows processing RTPs, which were not compressible at

[2]For the sake of simplicity one data channel is assumed and a max. scan-chain length of 4096 resulting in overall 27 scan-chains.

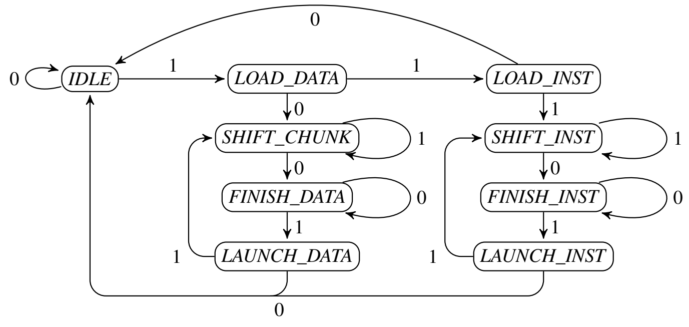

Figure 2: Simplified FSM of Bypass Controller

all in the past and, hence, had an adverse impact on the overall test compression. The proposed enhancement seamlessly combines a state-of-the-art technique with a lightweight codeword-based approach. To the end, the proposed architecture introduces only a negligible overhead in hardware and the computational effort for the retargeting procedure itself is manageable. First experiments already show that a data volume of RTPs can be significantly reduced by up 87.48% by at the same time no test coverage decrease.

Future work will focus on the extensive evaluation of the proposed technique for further industrial-sized designs to, among other, determine the best-matching dictionaries, e.g., by considering a reordering of the patterns, as well as determining the exact hardware overhead.

VI. Acknowledgment

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 276397488 – SFB 1232 in subproject P01 ‘Predictive function’.

References

[1] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, “Embedded deterministic test,” *IEEE Transaction on CAD of Integrated Circuits and Systems*, vol. 23, no. 5, pp. 776–792, 2004.

[2] A. Jas, J. Ghosh-Dastidar, and N. Touba, “Scan vector compression/decompression using statistical coding,” in *VLSI Test Symp.*, 1999, pp. 114–120.

[3] S. Mitra and K. S. Kim, “XPAND: an efficient test stimulus compression technique,” *IEEE Transaction on Comp.*, vol. 55, no. 2, pp. 163–173, 2006.

[4] S. Huhn, S. Eggersglüß, and R. Drechsler, “Reconfigurable TAP controllers with embedded compression for large test data volume,” in *IEEE International Symp. on Defect and Fault Tolerance in VLSI Systems*, 2017, pp. 1–6.

[5] F. G. Wolff and C. Papachristou, “Multiscan-based test compression and hardware decompression using LZ77,” in *IEEE International Test Conference*, 2002, pp. 331–339.

[6] W. R. A. Dias and E. D. Moreno, “Code compression using multi-level dictionary,” in *IEEE Latin American Symp. on Circuits and Systems*, 2013, pp. 1–4.

[7] S. Huhn, S. Eggersglüß, and R. Drechsler, “VecTHOR: Low-cost compression architecture for IEEE 1149-compliant TAP controllers,” in *IEEE European Test Symp.*, 2016, pp. 1–6.

[8] A. Wurtenberger, C. Tautermann, and S. Hellebrand, “Data compression for multiple scan chains using dictionaries with corrections,” in *IEEE International Test Conference*, 2004, pp. 926–935.

[9] H.-G. Liang, S. Hellebrand, and H. . Wunderlich, “Two-dimensional test data compression for scan-based deterministic bist,” in *IEEE International Test Conference*, 2001, pp. 894–902.

[10] C. Albrecht, “IWLS 2005 benchmarks,” 2005.

[11] H. Dhotre, M. Dehbashi, U. Pfannkuchen, and K. Hofmann, “Automated optimization of scan chain structure for test compression-based designs,” in *IEEE Asian Test Symp.*, 2016, pp. 185–190.

[12] S. Huhn, S. Eggersglüß, K. Chakrabarty, and R. Drechsler, “Optimization of retargeting for IEEE 1149.1 TAP controllers with embedded compression,” in *Design, Automation and Test in Europe*, 2017, pp. 578–583.