A Hybrid Embedded Multichannel Test Compression Architecture for Low-Pin Count Test Environments in Safety-Critical Systems

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

Daniel Tille§
§Infineon Technologies AG
85579 Neubiberg, Germany
Daniel.Tille@infineon.com

Rolf Drechsler*†
†Cyber-Physical Systems
DFKI GmbH
28359 Bremen, Germany

***Abstract*—This work presents a novel hybrid compression architecture that seamlessly combines the advantages of an embedded test compression technique with a lightweight codeword-based compression scheme. Embedded test compression has proven to be beneficial and is widely used in industrial circuit designs. However, particularly, in test applications within low-pin-count environments, a certain number of test patterns is incompressible and will, therefore, be rejected. This leads to a test coverage decrease which, in turn, jeopardizes the zero defect policy of safety-critical applications like automotive microcontrollers. Therefore, the rejected test patterns are typically transferred in an uncompressed way bypassing the embedded compression, which is extremely costly. The proposed hybrid architecture mitigates the adverse impact of rejected test patterns on the compression ratio as well as on the test application time of state-of-the-art techniques. The experimental evaluation of industrial-sized designs clearly shows that a significant compression ratio up to 67.4% and a test application time reduction up to 72.9% can be achieved when utilizing the existing multi-channel interfaces.**

I. Introduction

The latest accomplishments in the field of design and manufacturing 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 sophisticated on-board ICs to implement advanced driver-assistance systems. Due to the safety-critical aspect of such an application, different types of regulations, standards as well as practices have to be considered. Potential defects, which may have been occurred during the manufacturing process, have to be reliably detected to be compliant with these challenging requirements [1]. Thus, a very high test coverage is required leading to large sets of test patterns, which have to be executed during the manufacturing test. One further challenge concerns the available pin count. In industrial practise, the test is applied during different *test insertions*. The available infrastructure may vary significantly within the different environments. Dedicated burn-in devices, for example, provide only a very limited number of test pins. Other test insertions use heavy parallelization in order to reduce test costs, which again causes restrictions in terms of pin count.

Complex designs utilize powerful embedded test compression techniques like [2]–[8] to cope with this high test data volume. These techniques aim at reducing the overall testing time and, thus, saving costs. Typically, dedicated hardware is embedded on-chip that allows an on-the-fly decompression of the (compressed) test data during the transfer between the automatic test equipment and the circuit-under-test. Different hardware modules were developed in the past taking advantage of compression methodologies, which are well-known from the field of software compression. For instance, work [2] introduces a run-length encoding approach and work [4] invokes a static-encoding scheme. Other approaches utilize preconfigured dictionaries [7], extensive techniques like the *Lempel-Ziv* [5] algorithm or the Golomb-Code [8], which have been applied for large test data. However, these techniques introduce a significant hardware overhead and lack in a seamless integration into existing test data flows.

Commercially available state-of-the-art compression techniques like [6] 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 [9] 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. Techniques like [10] introduce conflict-aware test points to further increase the compacting abilities of the test set, even though this does not resolve the rejection of a single test pattern. Generally, the test pattern has to satisfy a large number of constraints coming from the compression architecture in order to be compressible. Since this is not always possible, these rejected tests lead to fault coverage loss. This effect is highly amplified in low-pin count test environments, which have been described above. In such cases, the number of incompressible patterns increases significantly. This is due to the fact that state-of-the-art compression solutions like [11] have the inherent characteristic that the compression effectivity is low if there are only very few pins, and therefore data channels, available. 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, i.e. bypassing the compression architecture. This yields to an adverse impact on the overall compression ratio as well as significant test cost increase.

This work[1] proposes a novel hybrid architecture, which seamlessly combines a state-of-the-art embedded test compression technique with a codeword-based compression scheme replacing the costly bypass structure. The codeword-based

[1] A preliminary version of this approach has been published in [12].

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 the basis of state-of-the-art formal techniques as proposed in work [13]. Besides that, the proposed architecture also allows the utilization of existing multichannel test compressions structures, which are typically introduced in industrial-sized designs. This allows reducing the transfer time of the RTPs even more and, hence, the resulting test application time. First experiments have already shown that a compression ratio up to 67.4% for the RTPs can be achieved. When considering existing multichannel structures, the final test application time can be reduced by up to 64.7%.

The structure of this paper is as follows: Section II briefly describes embedded test compression techniques including further approaches like codeword-based techniques. Section III draws the proposed hybrid compression architecture. The resulting hardware costs, as well as an architectural extension for multichannel usage, are described in Section IV. Experimental results are presented in Section V and, finally, Section VI summarizes the paper and provides an outlook on future works.

II. Background

Within the last decade, many different test compression techniques have been proposed in the literature, which all aim to reduce the test data volume and the test application time. This volume scales directly with the required memory resources on the automatic test equipment, which is one strictly limited resource, in particular, when considering test cost reduction techniques like multi-site testing [14].

A. Architecture of Embedded Test Compression

To take advantage of an embedded test compression technique, 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.

State-of-the-art techniques achieve a very high compression ratio, which mostly scales with the share of X-values in the processed test patterns. Thus, this compression ratio varies strongly depending on the structural properties of the data, e.g., most compression architectures put constraints on the distribution of scan cell assignments. A more critical circumstance is that a certain share of the overall pattern set is incompressible at all, because the generated test patterns do not satisfy the constraints of the compression architecture. Due to structural properties of these patterns, it is not possible that they are decompressed by $\mathcal{D}$ on-chip without any loss of information and, hence, they are rejected by the compression architecture. These RTPs have then to be transferred via a dedicated Bypass Controller $\mathcal{B}$ in an uncompressed and sequential fashion, which bypasses the $\mathcal{D}$-infrastructure completely. This results in a significant overhead regarding the test application time as well as the test data volume. However, these RTPs have to be executed to avoid any loss in test coverage, which is most important in domains in which a zero defect policy is required.

In principle, four components have to be implemented on-chip to instantiate such an embedded compression technique, which are as follows:

1) the Decompressor $\mathcal{D}$ restores the uncompressed test patterns, which are then loaded into the scan chains,
2) the Compactor $\mathcal{C}$ determines the test responses of the circuit by processing the scan chains' outputs,
3) the Bypass Controller $\mathcal{B}$ redefines the data path (including tail-gating all scan-chains) if the bypass is enabled for transferring uncompressed data (marked in red) and
4) a set of top-level pins holding a clock signal, two control signals (for $\mathcal{C}/\mathcal{D}$ and $\mathcal{B}$) as well as a data input and a data output channel for the compressed test pattern and test response.

Typically, multiple data channels are introduced when considering industrial-sized designs. This allows extending the available bandwidth to meet the requirements regarding the test application time. In turn, every additionally introduced data channel requires two further pins on the top-level, whose maximal number is strictly limited by, among others, the pad-frame design. Thus, this number has to be chosen per design individually since it forms a trade-off between data bandwidth and costs.

B. Codeword-based Compression

One important criterion of compression techniques concerns the completeness of the underlying compression algorithm. In this context, the completeness of the compression means that every arbitrary binary sequence can be processed. Consequently, none of the possible sequences, i.e., test patterns, will be rejected. By this, test coverage loss is prevented.

Powerful compression techniques, which are known to be complete, invoke statistical [15] or even codeword-based compression schemes [16]–[19] and are meant to be used for fully-specified, high-entropic data. For instance, the single bit injections are introduced into the compressed data to ensure the completeness of such a codeword-based technique with even small-sized embedded dictionaries [17]. The embedded dictionary holds a certain number of codewords, which depends on the actual dictionary size. More precisely, an embedded dictionary implements a mapping function ψ such that $\psi(c_i) \rightarrow u_i$ holds, i.e., every single (short) binary codeword c_i is projected to a (long) binary 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 codeword-wise 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. To further improve the compression effectiveness, the individual codewords can be configured dynamically prior to the data transfer.

III. Hybrid Compression Architecture

Embedded compression techniques reduce the test data volume significantly, which allows coping even with the large test sets of highly complex IC designs. However, a certain

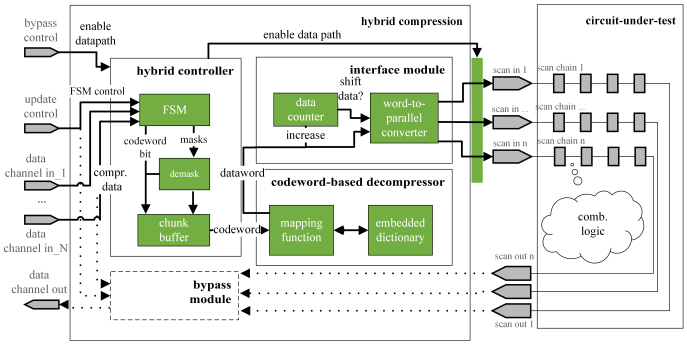

Figure 1: Hybrid Compression Architecture

share of the overall test patterns is rejected by the compression infrastructure leading to a shortcoming that jeopardizes the zero defect policy for safety-critical applications. The proposed hybrid compression architecture tackles this drawback by combining a state-of-the-art embedded compression technique with a codeword-based approach, which is meant to be applied for the RTPs exclusively. By this, the significant overhead introduced by the RTPs can be effectively reduced, while, at the same, keeping the fault coverage high.

A. General Idea

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.

A preliminary version of this approach has been published in [12]. The paper at hand presents new contributions such as the new technique of multichannel capabilities. We also provide a formal cost analysis of our proposed architecture and report on further conducted experiments.

Three different modules are required to implement the proposed hybrid architecture, as described in the following:

1) The *Codeword-based Decompressor* $\mathcal{D}_{CB}$ implements the embedded dictionary and provides an interface to the mapping function ψ,
2) the *Hybrid Controller* $\mathcal{H}$ controls the operations of the codeword-based components, which includes the newly introduced control signal *hybrid_en* that configures the external datapath, and ensures that the hybrid compression components behave transparently if not activated and
3) the *Interface Module* $\mathcal{I}$ implements the junction between the newly introduced codeword-based decompressor and the inputs of the scan-chains.

B. Codeword-based Decompressor

This module implements the codeword-based technique utilizing an embedded dictionary to decompress the RTPs on-chip without any loss of information. Reconsider that

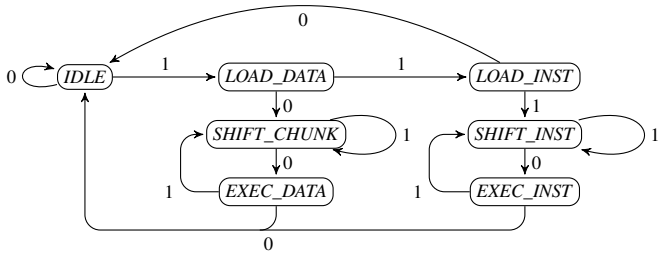

Figure 2: Simplified FSM of Hybrid Controller

these RTPs have been compressed once by the retargeting procedure prior to the test, i.e., the compressed data consists of a sequence of codewords $c_1, \ldots, c_n$. Methods to determine efficient codewords for given test data can be found in [20].

One integral component is the dictionary, which holds the codewords c_i in conjunction with the associated (uncompressed) dataword u_i. 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 [17]. 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. Besides this embedded dictionary, this module also implements the mapping function $\psi(c_i) \rightarrow u_i$, which provides the functionality for the later decompression.

C. Hybrid Controller

This module implements the necessary control structures by introducing a *Finite-State-Maschine* (FSM), whose state transitions are controlled by the external compression control signal and synchronized with the test clock. The FSM design, which is shown in Figure 2, allows differentiating between data and instruction (*inst*) branch, which allows a clear distinction between data- and control-path. The current implementation covers four different instructions and, consequently, these instructions are encoded by opcodes holding two bits. The supported instruction are as follows:

- ‘00’ *inactive*; $\mathcal{H}$ remains bypassed,
- ‘01’ *preload*; the embedded dictionary can be configured with specific codewords, which are adopted to the data,
- ‘10’ *compress*; codeword-based compression is enabled (disabled) and prepared for decompressing incoming RTP, which is also indicated by setting the signal (*hybrid_en*) that controls the external datapath and
- ‘11’ *debug*; offers trace data for debug purposes.

One important criterion of the overall design of $\mathcal{H}$ is about the datapath of $\mathcal{H}$, which remains completely isolated until the bypass control signal is set. This principle allows avoiding any unintended interference when the regular compression infrastructure decompresses the unrejected test patterns.

Loading instructions: The FSM remains in the *IDLE* state as long as the bypass and compression control signal are

both set. To execute an instruction, the specific opcode has to be bit-wise transferred to $\mathcal{H}$ via the data input channel while recursively traversing the state *SHIFT_INST* twice. This state can be reached by passing both *LOAD_DATA* as well as *LOAD_INST* states first. After the transfer is completed, the transferred opcode is decoded and, finally, executed within the state *EXEC_INST*. This instruction cycle ends up in the *IDLE* state or at shifting a further instruction. If the instruction *compress* has been executed, the newly introduced *hybrid_en* signal is set. This signal controls the external datapath with respect to the inputs of the scan-chains.

Loading data: The compressed RTP consists of a sequence of codewords $c_1, \ldots, c_n$. Each and every of the n codewords has to be transferred in a bit-wise fashion. Analogously to the instruction load, the bit-wise transfer for the i-th codeword c_i is executed while recursively traversing the state *SHIFT_CHUNK*. A bit of a codeword (*chunk*) is transferred with each iteration and stored in a dedicated chunk buffer. The state is changed to *EXEC_DATA* as soon as all bits of the current codeword c_i are completely transferred. The actual decompression is executed within this state by orchestrating the mapping function ψ. This state separation allows identifying the end of a single codeword uniquely. The next reached state is then either *SHIFT_CHUNK* to start a further codeword transmission or *IDLE* to finish.

D. Interface Module

This module realizes the junction between the introduced $\mathcal{D}_{CB}$ and the N inputs of the scan-chains. In this scenario, $\mathcal{D}_{CB}$ 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 especially 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. If enough data bits are available, the parallel shift operation too all scan-chains is performed or, otherwise, more data bits are received until the required amount of data is available. Thus, three different components are required to implement $\mathcal{I}$ 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* (CR) storing the currently decompressed bits and
- an *update register* (UR) 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 and stored in CR. 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 CR to UR, which is then used as the data source. The counter is then decreased by N.

E. Exemplary Application

The remainder of this section describes the exemplary application of the hybrid compression technique. For the later example, it is assumed that the components of the compression architecture as well as of the hybrid compression have been both implemented in the circuit and the test pattern set has been generated. Furthermore, it is assumed that the RTPs have been identified and retargeted by the utilized retargeting framework.

The utilization of the hybrid compression is as follows (consider t as the point of time in sense of passed cycles):

1) t=0: The external bypass control signal is set to '1' and remains in this state, which activates the datapath of $\mathcal{H}$.
2) t=1..3: The external update control signal is set to '1' for three clock cycles to, finally, reach the state *SHIFT_INST*.
3) t=4..6: The opcode '10' is shifted into $\mathcal{H}$ by controlling the data input channel to '1' (t=4) and to '0' (t=5). The transferred instruction is decoded and executed in the state *EXEC_INST* (t=6). This implies that the *hybrid_en* signal is set, which activates the datapath between $\mathcal{I}$ and the input of the scan chains and, thus, deactivates the classic bypass architecture.
4) t=7..9: The update control signal is driven such that the state *SHIFT_CHUNK* is reached.
5) t=10..11+x: Transfer of the first codeword, which requires at least one but up to three cycles depending on the number of chunks. The data bits are transferred via the data input channel.
6) t=12+x..13+x: The first codeword c_i is completed, which is indicated by the state transition to *EXEC_DATA* (induced by the control signal change '1' to '0'). The decompression is performed by invoking $\psi(c_i)$ to restore the dataword u_i with respect to the embedded dictionary. Subsequently, u_i is directly stored in the capture register and the data counter is increased by $|u_i|$.
7) t=14+x: If a further codeword c_{i+1} has to be transferred, the state is changed to *SHIFT_CHUNK* and it is continued with Step 5). Otherwise, the state is changed to *IDLE*. The complete datapath of $\mathcal{H}$ is isolated by applying Step 1) in reverse.
   In parallel, the value of the counter of $\mathcal{I}$ is evaluated: If counter $\geq N$ is valid, the least recently received N bits are moved from the CR to the UR such that a parallel shift operation is performed (t=15+x), Furthermore, the counter is decreased by N.

As stated, the individual codewords of the embedded dictionary can be configured by first executing the *preload* instruction [cf. Step 2), 3)] and, secondly, the datawords have to be transferred in a bit-wise fashion - analogously to Step 4) to 6). To keep the exemplary application short, this configuration phase is not discussed in more detail.

IV. Cost Analysis and Multichannel Utilization

Important aspects concern the estimated hardware overhead in sense of additional registers, which are introduced by the hybrid compression approach within an arbitrary design, and the measurable benefit regarding cost savings in sense of tester memory or testing time. The remainder of this section discusses both of these aspects.

A. Hardware Costs

The hardware costs of $\mathcal{D}_{CB}$ and $\mathcal{H}$ are both completely independent from the design size but are mainly determined by the parameters of the embedded dictionary. These parameters are the *Maximal Codeword Length* (MCL) defining the upper bound for unique binary encodings of the code words and the *Maximal Dataword Length* (MDL) determining the word size of the single dictionary entry. More precisely, the dictionary holds two entries with a length equal to 1, four entries with a length equal to 2 and eight entries with a length equal to 3 etc. Generally spoken, **(a)** $\sum_{i=1}^{MCL}(2^i \cdot i)$ bits are required to store all codewords. Furthermore, the associated dataword (with a maximal length of MDL bits) for all possible codewords has to be considered with **(b)** $\sum_{i=1}^{MCL}(2^i \cdot MDL)$ bits.

In contrast to this, the costs of $\mathcal{I}$ scale with the number of considered *inputs of scan-chains* (N). The size of UR has to be equal to **(c)** N since this is the exact number of bits which are required to perform the parallel shift operation. The received and decompressed bits are stored in CR, which has to hold a size of **(d)** $N+MDL-1$ bits. This covers the scenario that CR is filled with $N-1$ data bits (the critical amount for performing the parallel shift operation is not yet reached) and a further dataword of MDL. Additionally, $\mathcal{I}$ contains an (unsigned) data counter of **(e)** $\lceil \log_2(N+MDL-1) \rceil$. Besides this, some extra registers are required to hold the current state and the loaded instruction of $\mathcal{H}$ and, finally, chunk-buffer holding **(f)** MCL bits. Accumulating (a) to (f) results in

$$\sum_{i=1}^{MCL} [2^i \cdot (i+MDL)] + \lceil \log_2(N+MDL-1) \rceil + 2N + MDL + MCL - 1 \quad (1)$$

with MCL='3' and MDL='8'

$$2N + \lceil \log_2(N+7) \rceil + 156 \quad (2)$$

As shown in (2), the hardware cost scales super-linearly with the number of scan-chains. In industrial-sized designs with a huge number of scan-chains, the individually input is not accessible but multiple scan-chains are tail-gated. This heavily reduces the number of scan-chain inputs, which have to be directly driven. For the sake of simplicity, this technique has not been considered in this work.

B. Multichannel Transfer

As described in the exemplary application in Paragraph III-E, the transfer of all codewords $c_1, ..., c_n$ is implemented in a sequential fashion. More precisely, the chunks of each codeword c_i are transferred, whereby each of them contains up to *MCL* (here 3) chunks. Thus, it is iterated over the state *SHIFT_CHUNK* up to *MCL*-times. This number of required iterations can be statically reduced to 1 by utilizing existing multichannel structures, which typically exists in large designs anyway. To achieve this, $MCL+1$ channels are orchestrated to transfer all chunks in parallel plus one additional bit implementing a masking scheme. Considering these channels as $MCL+1$-bit-wide bus $data[MCL:0]$, the most-significant bit $data[MCL]$ indicates whether all the remaining $data[MCL-1:0]$ are used to transfer a codeword of length MCL. If $data[MCL]$ is not set, the $data[MCL-1]$ acts as this mask for the remaining $data[MCL-2:0]$, indicating a codeword of length $MCL-1$. It is assumed that at least $data[0]$ holds a chunk of a codeword with length 1. This scheme allows a further reduction regarding the test application time, though, it introduced a further bit to be set when applying the masking operation, which results in a loss of compression. However, during test the test time is generally more critical than the test data volume.

TABLE I: Benchmark Circuits

circuit	#scan. elements	#scan-chains	#channels
ethernet	10,038	5	1
vga_lcd	12,983	14	1
leon3	82,251	81	4
netcard	96,569	97	4

V. Experimental Evaluation

This section describes the experimental evaluation of the proposed technique, which has been done using different industrial-representative *OpenCores* as well as *Gaisler Research Benchmark* circuits from the *IWLS 2005* benchmark collection.

The underlying scan capabilities as well as the embedded test compression has been inserted by a commercial tool. The characteristics of the resulting circuits are shown in Table I as follows: The benchmark circuit name, the overall number of scannable sequential elements included in the design (#scan. elements), the number of introduced scan-chains[2] (#scan-chains) and the number of data channels being accessible on the top-level (#channels). This ratio of scan-chains divided to introduced channels has a strong impact on the share of rejected test patterns. Reconsider that certain applications like the low pin count test holds a very high ratio. 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. 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 hybrid 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 [20], which has been slightly adopted. Among others, it is combined with a partitioning approach of work [13] to keep the run-time manageable while further improving the effectiveness. Hence, data blocks with a size of 8k 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 hybrid compression module.

Table II presents the detailed results of the conducted experiments. More precisely, the benchmark circuit name, the overall number of test patterns, which have been generated by the commercial tool (#pattern), the number of required data blocks per RTP, the minimal, average and maximal retargeting run-time per pattern in seconds are shown. Furthermore, Table II also presents the minimal, average and maximal pattern compression ratio in percent and, finally, the further achieved test time reduction in percent - both compared against

[2]Note that a maximal scan chain length of 1024 elements is considered.

TABLE II: Benchmark results

circuit name	#pattern	#blocks	retargeting run-time [s]			pattern compression ratio [%]			test time red. [%]			multichannel test time (volume) red. [%]		
			min.	avg.	max.	min.	avg.	max.	min.	avg.	max.	min.	avg.	max.
ethernet	1,049	2	13.2	26.4	47.2	31.1	56.4	79.3	34.6	55.5	72.8	n/a	n/a	n/a
vga_lcd	1,286	2	13.6	37.6	42.1	30.6	40.7	56.1	30.5	41.5	47.9	n/a	n/a	n/a
leon3	6,332	11	64.3	293.7	513.2	31.7	48.7	79.2	27.2	48.8	69.4	38.4 (14.8)	59.4 (48.1)	84.8 (77,9)
netcard	9,939	12	104.4	126.0	230.4	28.5	38.1	67.4	29.6	45.7	65.7	34.6 (20.1)	51.2 (39.8)	72.9 (48.9)

the regular bypass transfer without any compression at all - and, finally, the minimal, average and maximal achievable test time (volume) reduction in percent when considering the multichannel interface – if available and, otherwise, a 'n/a' is shown.

Note that the configuration of the embedded dictionary is applied once prior to the data block transfer. Both, the configuration time as well as configuration bits are included in the shown benchmark results. Furthermore, the proposed hybrid compression module is complete by construction, i.e., any arbitrary RTP can be transferred and, thus, no test coverage loss is introduced. In case of the *netcard* benchmark, at first, the experiments clearly show that the run-time of the retargeting engine, while considering different design sizes, is stable **per** block. 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.

Besides this, the achieved compression ratio is stable over the conducted experiments as well, which is indicated by a variance standard deviation of 6.4% (*netcard*). In contrast to this, the deviation between the minimum and the maximum of achieved test time reduction is greater, which is due to the fact that this depends on the distribution of codewords with a length of 1, 2 or 3, respectively. Thus, the test time reduction is not directly connected with the compression ratio. Latter is determined mainly by the associated datawords the introduced codewords are pointing to. When processing the largest *netcard* circuit with the greatest number of patterns ($N = 9939$), the resulting test data volume can be significantly compressed by 38.1% on average in conjunction with a test time reduction of 45.7% on average. By invoking the proposed multichannel scheme, the test application time can be further reduced by up to 72.9% while still achieving a compression ratio up to 48.9%, which exhibits a large potential of saving test costs.

VI. Conclusions

This paper proposed a hybrid architecture, which seamlessly combines a state-of-the-art embedded compression technique with a lightweight codeword-based compression scheme to enable the processing of rejected test patterns. So far, these rejected patterns had an adverse impact on the overall compression effectiveness and, hence, untapped potential of reducing the test costs remained. The proposed architecture introduces only a negligible hardware overhead in sense of register count to the design and the computational effort for the preprocessing step, i.e., the retargeting, is manageable. Different experiments on industrial-representative designs – with up to approximately 100k flip-flops – demonstrate the effectiveness of the proposed hybrid architecture. A significant test data volume reduction of up to 67.4% was achieved and, most important, without any loss in test coverage at all. Furthermore, the test application time was reduced even stronger by up to 64.7%. In addition to this, the proposed hybrid architecture allows utilizing existing multichannel structures. By this, it was possible to further decrease the test application time by up to 72.9% with up to 48.9% test data volume reduction. Future work will focus on determining the best-matching dictionaries with respect to the possibility of pattern reordering.

VII. 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] M. Weber and J. Weisbrod, "Requirements engineering in automotive development-experiences and challenges," in *IEEE Joint International Conference on Requirements Engineering*, 2002, pp. 331–340.

[2] A. Jas and N. Touba, "Test vector decompression via cyclical scan chains and its application to testing core-based designs," in *IEEE International Test Conference*, 1998, pp. 458–464.

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

[4] V. Iyengar, K. Chakrabarty, and B. Murray, "Deterministic built-in pattern generation for sequential circuits," *Journal of Electronic Testing*, vol. 15, no. 1-2, pp. 97–114, 1999.

[5] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transaction on Information Theory*, vol. 23, no. 3, pp. 337–343, 1977.

[6] 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.

[7] L. Li and K. Chakrabarty, "Test data compression using dictionaries with fixed-length indices," in *VLSI Test Symp.*, 2003, pp. 219–224.

[8] A. Chandra and K. Chakrabarty, "System-on-a-chip test-data compression and decompression architectures based on Golomb codes," *IEEE Transaction on CAD of Integrated Circuits and Systems*, vol. 20, no. 3, pp. 355–368, 2001.

[9] 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.

[10] C. Acero, D. Feltham, Y. Liu, E. Moghaddam, N. Mukherjee, M. Patyra, J. Rajski, S. M. Reddy, J. Tyszer, and J. Zawada, "Embedded deterministic test points," *IEEE Transaction on CAD of Integrated Circuits and Systems*, vol. 25, no. 10, pp. 2949–2961, 2017.

[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, D. Tille, and R. Drechsler, "Hybrid architecture for embedded test compression to process rejected test patterns," in *IEEE European Test Symp.*, 2019, pp. 1–2.

[13] 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.

[14] V. Iyengar, S. K. Goel, E. J. Marinissen, and K. Chakrabarty, "Test resource optimization for multi-site testing of SOCs under ATE memory depth constraints," in *IEEE International Test Conference*, 2002, pp. 1159–1168.

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

[16] 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.

[17] 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.

[18] 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.

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

[20] 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.