A new Compression/Decompression method for non Correlated Test Patterns: Application to Test Pins Expansion*

Walid Maroufi
LIP6 Laboratory
Couloir 55-65, 4 Place Jussieu
75252 Paris Cedex France
Tel. (+33)1 44 27 71 23 - Fax. (+33)1 44 27 72 80
E-mail : Walid.Maroufi@lip6.fr

Abstract

This paper presents a new statistical lossless compression/decompression method which aims at reducing the amount of test data specially in case of feeble correlation. This obviously has the advantage of reducing test time and test storage problems. The method is implemented within our CAS-bus SoC test architecture which is controlled through an IEEE1149.1 TAP controller. Without loosing the IEEE1149.1 compatibility, the TAP is slightly modified in order to control the decompression process.

1 Introduction

Testing systems-on-a-chip induces many challenges like defining adequate new test architectures, standardizing core test wrappers (P1500) and developing new software test tools for SoC design for testability as well as for test plans or vectors generation. One of the most important problems is the transfer of test data from testers to the internal cores [1]. This is normally resolved by the SoC Test Access Mechanism (TAM) [2]. But, sometimes, some constraints can degrade the TAM performances. The one we focus on in this paper is when The TAM width is greater than the SoC test pins number(figure 1).
In [3], we present a P1500 compatible reconfigurable TAM architecture (CAS-BUS). It was decided that this TAM architecture will be controlled through a TAP mechanism (figure 3) with multiple TDI/TDO pins pairs. This architecture is confronted to the problem we depicted above when the number of the TDI/TDO pairs is less than the SOC internal test data bus width. In such a case, an expansion of one or more TDI/TDO pairs is necessary in order to connect all test bus wires (figure 2). This will obviously increase test time and so degrade test performances, since we have to transfer from one TDI/TDO pair, two or more serial test data chains. Of course, a good distribution of core internal scan chains on the SoC global test chains can limit test performance degradation, but it can't represent the best solution in any application case. To cope with this problem, we present in this paper an expansion mechanism, based on a compression/decompression method, which aims at limiting test time increase due to a simple expansion. This novel compression/decompression method, which has a compression degree between 25 % (minimum average) and 50% (maximum) is primarily dedicated to our architecture (CAS-bus + TAP control) but it can be adapted to other architectures. The 25 % minimum average gain is completely independent from any test data correlation. The choice of

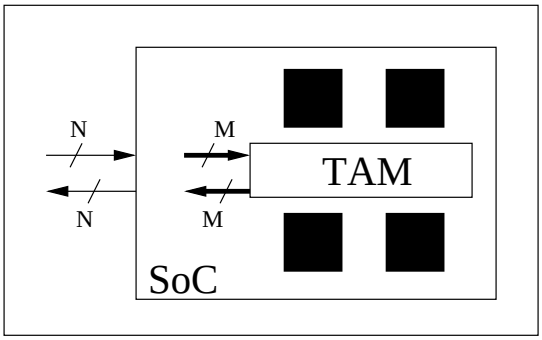

Figure 1: problem: N < M

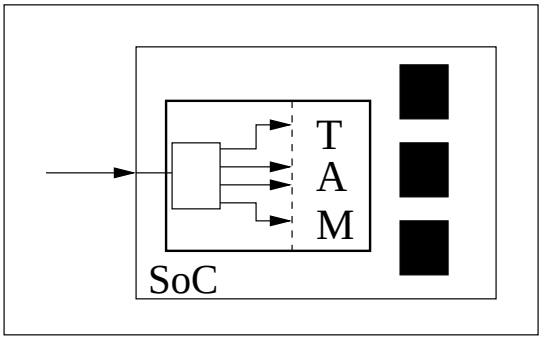

Figure 2: Expansion

*This work has been partly supported by MEDEA AT403-SMT Project

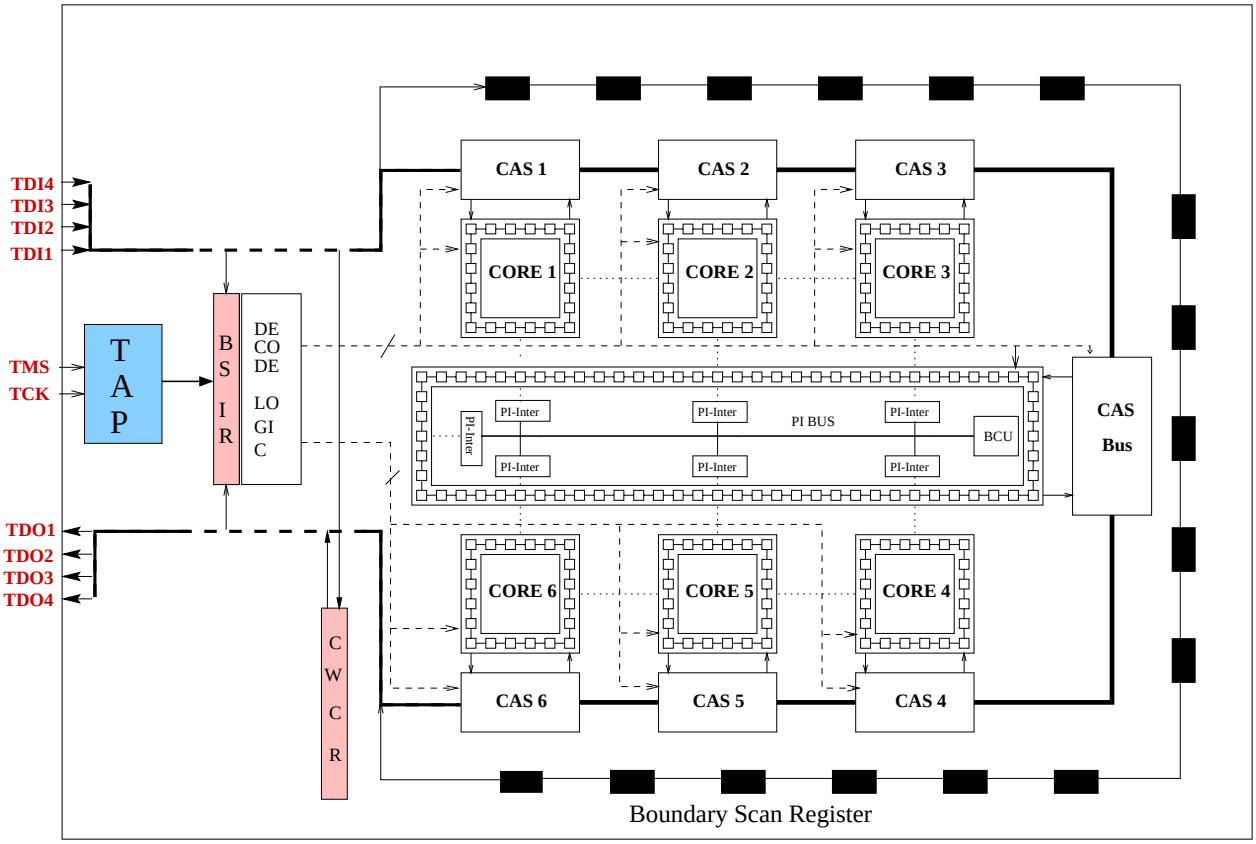

Figure 3: CAS-bus/TAP architecture

other compression/decompression statistical coding methods like those based on Huffman or similar coding [4] or test vector decompression via cyclical scan chains techniques [5] was studied, but their dependence on test data correlation represents an important shortcoming.

This paper is organized as follows: Firstly we will briefly present in section 1 the CAS-bus/TAP architecture and meanly it is controlled through a TAP port. The compression/decompression technique is described in section 3. The way by which the compression/decompression module is controlled through the TAP mechanism, without requiring any additional test signals, is presented in section 4. An experiment result is available in section 6.

2 CAS-bus with TAP control

The reconfigurable CAS-bus architecture we propose in [3] is characterized by its flexibility, scalability and dynamic aspect. It consists mainly of Core Access Switches (CASes) connected to each other through a test bus. A CAS is a simple programmable controller connected to each testable core through a P1500 or compatible wrapper. Test bus is a set of wires transporting test data through the SoC. Depending on its configuration, each CAS chooses P wires, from the N composing the test bus, to connect them to the test pins of the core. The (N-P) remaining wires bypass it. Thus, after a step of global configuration of CASes, one test bus wire will represent a test chain composed of multiple core local test chains (scan chain, BIST initialization register, etc...).

The TAP mechanism has been chosen to control the architecture (figure 3). Three new mandatory Boundary Scan instruction have to be added:

- The **CASWrapper_coupling** instruction which serves to initialize the CWCR register. Each bit of the CWCR register is dedicated to a pair Wrapper/CAS and indicates if, during a SoC global configuration step, Wrapper instruction register and CAS instruction register are connected to each other.
- The **CAS_config instruction** which connects TDI1/TDO1 to one test bus wire connecting itself CASs and Wrappers (depending on CWCR register value) instruction registers to each other. This is the step of SoC global configuration. Data transferred through the test bus wire represents the chosen structure for the SoC test architecture.
- The **CAS_test** which connects TDI/TDO pairs to the test bus wires in order to transmit test vectors to the core's internal test chains and pick up responses.

Of course, a classic Boundary scan register for the SoC is planned.

3 Proposed Compression/Decompression Method:

The method we propose is a lossless statistical one which treats 4 bit-length samples of the test data. The choice of 4 bit-length samples will be explained at the end of this paragraph. To ease the explanation of this method, let's

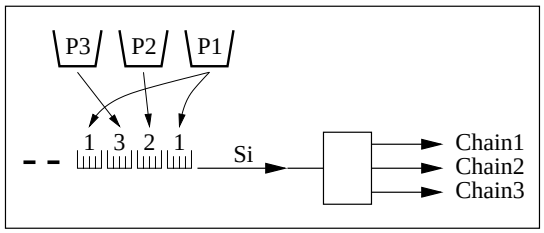

Figure 4: Example of expansion

take the expansion example shown in figure 4. It consists of three scan chains which must be accessed through only one serial input (**si**). P1, P2 and P3 represent respectively the global length of test patterns for chain1, chain2 and chain3. We suppose a cyclic transfer of the 4 bit-length samples coming on **si** to the three chains. Without any compression of test data, it was clear that the total test time needed to transfer all test patterns is equal to $(P1 + P2 + P3)$ cycles. This represent a large test time, knowing that if we had 3 serial inputs, the total test time can be reduced to max(P1,P2,P3) cycles (parallel testing of the three chains). For 4 bit-length samples, the num-

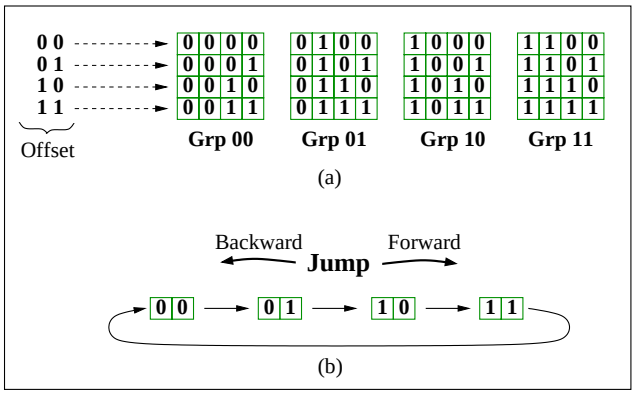

Figure 5: Groups example

ber of possible combinations is of course equal to 16. We arbitrary classify the 16 combinations through 4 groups of 4 combinations each (example: figure 5-a).
Within one specific group, each combination is normally coded through 2 bits (offset of the combination in the group). Groups are also coded through two bits, but we consider another way to select them. Indeed, we consider that they are cyclically chained (figure 5-b), so there are three possibilities to change from one group to another:

- One forward jump in the cyclic group chain.
- One backward jump.
- Two backward (or forward) jumps.

Thus, the idea is to start from one chosen group (for example group 00) and to code samples through two bits. If any group modification is necessary, a dedicated signal (for example **Grp_chg**) must indicate it. When the **Grp_chg** signal is active, the value at the input **si** indicates the direction of the jump (1: forward jump, 0: backward jump). In case of two jumps, the signal **Grp_chg** must stay active during two clock cycles while the value at the input **si** must be also stable (at 0 or at 1).
The gain for each 4bit-length sample can be estimated as follows:

- case of no jump : 2 bits (2 cycles) for the coding $->$ 50 % gain.
- case of one jump : 2 bits (2 cycles) for the coding and one cycle for the jump indication $->$ 25 % gain.
- case of two jumps : 2 bits (2 cycles) for the coding and two cycles for the jumps indication $->$ 0 % gain.

When no special correlation is present in a test data amount, the percentage of jumps can be statically seen as: 25 % of no jumps, 25 % of one forward jump, 25 % of one backward jump and 25 % of two jumps. So the global average gain can be calculated as : (0.25 x 50 %) + (0.25 x 25 %) + (0.25 x 25 %) = 25 %. Of course this gain is not static and essentially depends on the number of jumps within a test process. The maximum gain that we can reach is when no jump is necessary and is equal to 50 %.
One way to limit the number of jumps, is to modify the distribution of the 16 main combinations through the 4 groups. It is easy to see that for each test process, an optimal distribution can be found.
It is clear that to limit the number of possible jumps, the number of groups must be small. Suppose now that we use a 6 bit-length samples, the number of possible combinations will be 64. Distributed within four groups (for example), we will obtain 16 combination per group: 4 bit for coding offsets. In case of no jumps, the gain will be equal to (6-4/6)=33 % which is less than gain obtained for 4 bit-length samples. When distributed through more groups, 16 for example, offsets will be coded through 2 bits but the number of jumps will be very important. Suppose for example that we must make a jump of 8 groups, so 8 clock cycles must be added to the offset 2 cycles: then we have 10 cycles to code a 6 bit-length sample (-66 % gain !). The same problems occur for larger samples (8,10, etc...), that's why we decided to use four bit-length samples

4 Integration within the CAS-bus/TAP architecture

As
said in the introduction, the compression/decompression method we present is first dedicated for the CAS-bus architecture controlled using the 1149.1 TAP presented in section 1. The serial input is obviously TDI, and in order to avoid additional signal we decide to use the TMS signal to indicate a jump. An expansion/decompression module is added to the architecture as well as an MISR from which a signature can be shifted out through TDO (figure 6). In addition to the three new instructions pre-

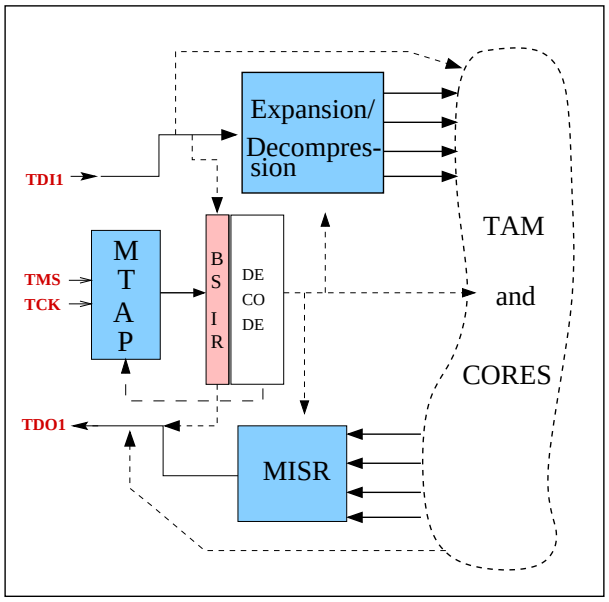

Figure 6: CAS-bus/TAP architecture with expansion/decompression module

sented in section 1, a fourth instruction called **DECOMPRESS_test** replaces **CAS_test** when transferring test vectors. This instruction connects the TDI serial input to be expanded to the expansion/decompression module and the TDO output to the MISR.
It is important to note that only one clock cycle is allowed to indicate a jump (2 cycles for two jumps). Thus, using the TMS signal to indicate group jumps with a classic standard TAP controller is impossible. Indeed, when TMS is activated within a test vector transfer, it is impossible to go back to the **shift-DR** state before 3 clock cycles, which does not suit to our constraints.
The solution we propose is to use a modified TAP which we call MTAP. This TAP behaves exactly like a classic standard 1149.1 controller except when a **DECOMPRESS_test** is shifted in. In that case, two supplementary states (**ONE-jump** and **TWO-jumps**) are added to the data register control part of the TAP (figure 7). This is automatically done when the instruction register decoder decodes the **DECOMPRESS_test** instruction. The main advantage of MTAP is that the constraint of one clock cycle for one jump is satisfied. Indeed when TMS signal is activated within a test vector transfer, MTAP moves to the state **ONE-jump** (the direction of the jump depends on the value of TDI). If another jump is necessary, TMS must stay at one and MTAP moves to the state **TWO-jumps**, otherwise it goes back to a shift state: **shift-DR**.
Figure 8 shows an expansion/decompression sequence of one TDI input to three serial chains. **CHi-VALj** represents a 4 bit-length sample which is part to chain i. **CHi-CDj** indicates the compressed code of **CHi-VALj**. The four groups are named A, B, C and D and the register **Current-group** is initialized to B. Once three codes (ex : CH1-CD1,CH2-CD1 and CH3-CD1) are shifted in, the decompressed values (CH1-VAL1,CH2-VAL1 and CH3-VAL1) are shifted out through the three chains. It is important to note that the **Pause-DR-bis** and **Shift-DR-bis** signals replace the initial **Pause-DR** and **Shift-DR** signals for the three chains control. However, the initial signals (**Pause-DR** and **Shift-DR**) still control the expansion/decompression module serial internal chain.
Since 12 clock cycles are involved to shift three 4 bit-length samples without any compression/decompression process, it is simple to see the gain obtained for each sequence: 3 cycles for CHi-VAL1 and 2 cycles for CHi-VAL2.

5 Implementation

In order to clock the feasibility of the method, we implemented a cycle precise synthesizable description of the decompression module, the MISR, the MTAP and all the IEEE 1149.1 remaining parts. The modular description was synthesized through SYNOPSYS design analyzer. Simulations coincide well with expected results.
In the synthesized description, groups combinations can be modified during the step of global configuration. Indeed, a (16x4)bits internal register is chained with CASs instruction ones. Of course, it is possible to do that differently, depending on the designer constraints.
The decompression module was synthesized in a fix configuration, which means that the number of serial output was fixed (4 outputs). However, it is possible to imagine a reconfigurable module with a scalable number of derived chains. In its fixed version, the module contains 40 flip-flops (without counting the groups flip-flops) and less than 500 combinational gates, which doesn't represent an important area overhead compared to the SOC global area.

6 Example

In order to experiment the method with a real test example and with the lack of real SOC application, we decided to apply it to the test patterns of a real circuit done in our laboratory. The circuit, named PCIDDC, is a 200K

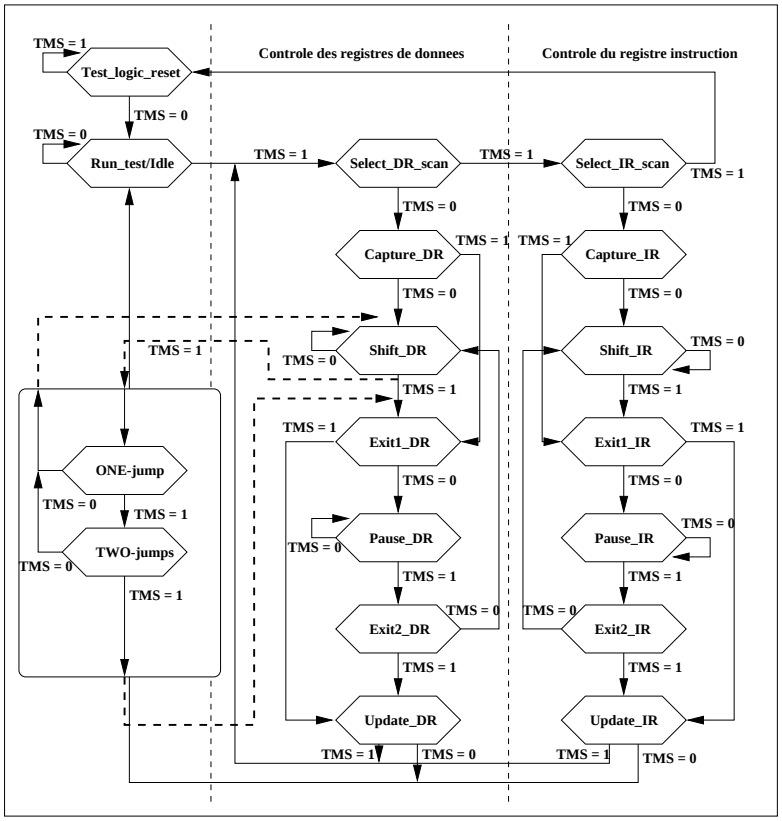

Figure 7: Modified TAP state machine

transistor, Network interface Component for PCI bus including 4 scan chains with a 1028 total scan length.
Using our method, we reach easily 35 % of compression rate by randomly modifying the distribution of combinations within groups. A software module is dedicated to that and the 35 % rate was obtained after few processing minutes.
For the same patterns, we calculated the percentage of appearance of each combination in order to prepare a Huffman coding:

- For 4 bit-length samples : all the 16 combinations have a close appearance degree (between 5.3 % and 8 %).
- For 6 bit-length samples : all the 64 combinations have a close appearance degree (between 1 % and 1.9 %).

This means that correlation between samples is very loose and no important compression can be reached with a Huffman coding. Obviously, for all the methods based on similar coding, the problem remains the same, since they all necessitate a good correlation between samples.

7 Special cases

In this paragraph we present two special cases which we think are the most important ones.

First, is it possible to use the method without expansion (one serial input and one internal chain) ? This is possible, but two test clocks are necessary: one slow clock for the compressed patterns (Tester) and the second, a fast clock, for the decompressed chain (figure 9). The same technique is used in [4]. Note that this is not necessary in case of an expansion, since the time needed to input compressed codes is at least equal to 2 N cycles (N is the number of target internal chains, N >= 2), which is enough to shift in parallel 4 bit-length samples. Second, how the method

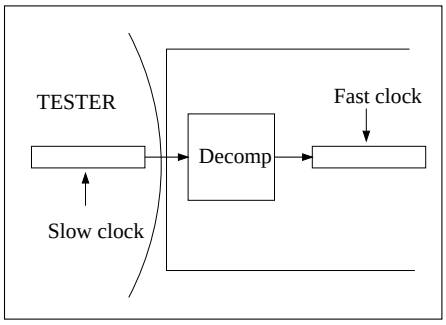

Figure 9: Case of no expansion

can be used when more than one serial input is available? For such a case, a simple rule is to minimize the number of chains to be connected to the expansion/decompression module. So, for N serial inputs and M internal chains, we

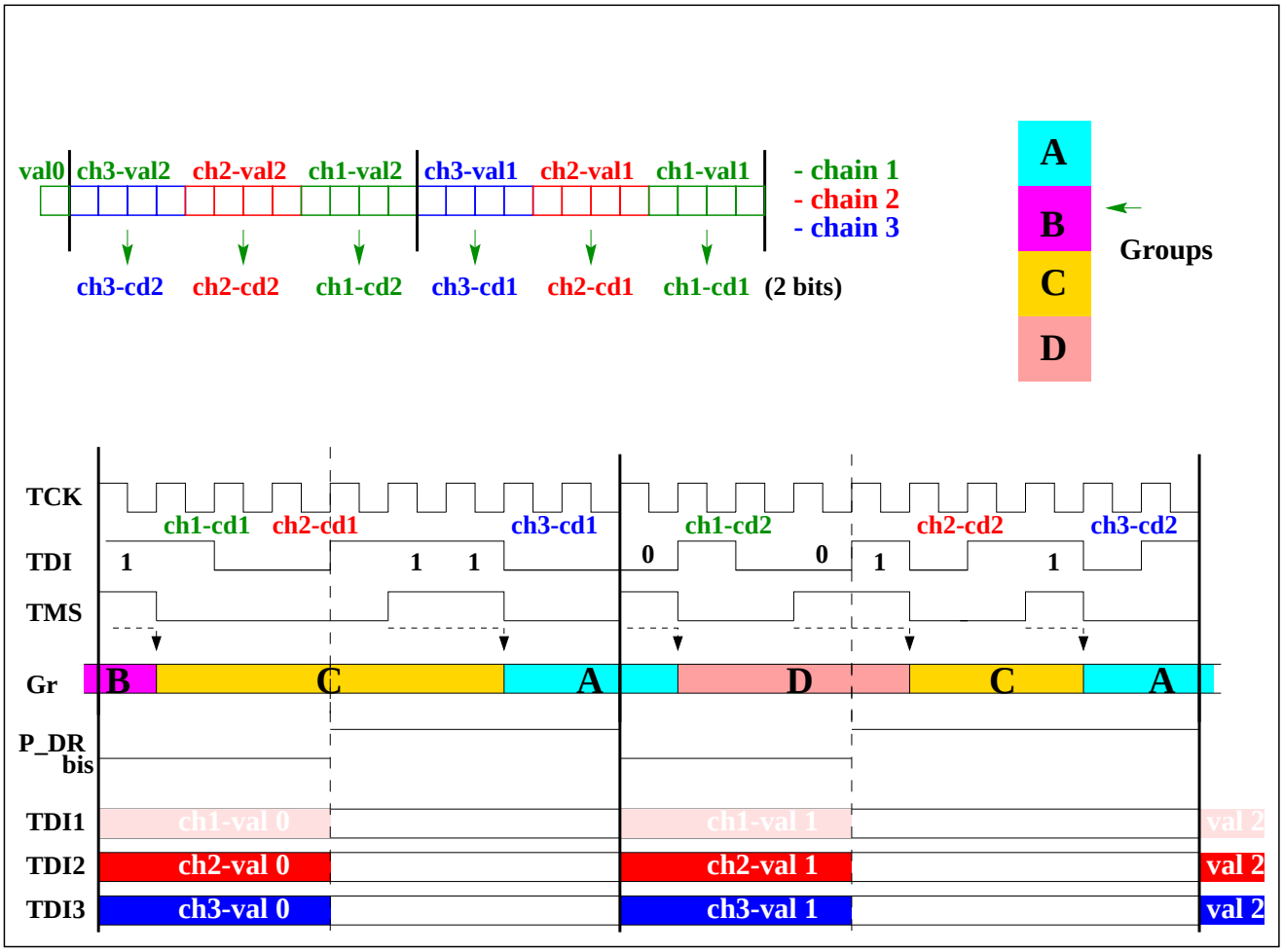

Figure 8: Expansion/Decompression sequence

can connect (N-1) inputs to (N-1) chains and use the last input to expand the remaining (M-(N-1)) chains. The distribution of chains on the direct or expanded serial inputs must be a result of a global tradeoff taking into account test chains length and relative test data amount. Using more than one decompression module is possible, but this necessitates more than one MTAP mechanism.

8 Related software development

Around this method, we are developing some tools which aim at:

- evaluating the compression degree for a given test expansion/decompression process.
- making the tradeoff concerning which internal chains can be connected to the decompression module and which can be directly connected to serial test inputs.
- verifying if it's profitable to change groups combinations within a same test process in order to enhance compression degree.

All these tools are to be merged in the same package of tools which we are developing around the CAS-bus/TAP architecture.

9 conclusion

We present in this paper a compression/decompression method which can be integrated within the CAS-bus/TAP architecture. Even if it doesn't give spectacular compression performances with correlated test patterns like Huffman/similar or run-length coding methods, it is very useful with non correlated test patterns. Compression degree varies between 25 % (minimum average) and 50 % (maximum).
Particularly, we propose this method to enhance test performances in term of test time when expanding a test serial input to more than one SOC internal test chain. An expansion/decompression sequence example has been presented.
A simple modification has been introduced in the IEEE1149.1 TAP controlling the CAS-bus architecture. The modified TAP (MTAP) behaves exactly as a classic

TAP, except when it's an expansion/decompression process. The functionality of the global architecture: CAS-bus + MTAP + Decompression Module + MISR has been validated at gate level.
In another hand, the method has been tested on the real test patterns of a circuit made in our laboratory (PCIDDC: a Network Interface Component) and 35 % of compression degree is reached. The use of other compression methods was studied for the same patterns, but due to their feeble correlation, compression degrees were not significant.
Some dedicated tools are being developed, they will be included within the CAS-bus/TAP architecture tools package.

Acknowledgments

The auther is grateful to his colleagues **Mounir Benabdenebi** and **Mereyem Marzouki**, for the development on CAS-bus architecture and its control.

References

[1] Y. Zorian. Test requirements for embedded core-based systems and ieee p1500. In *International Test Conference*, pages 191–199, 1997.

[2] E. J. Marinissen, Y. Zorian, R. Kapur, T. Taylor, and l. Whetsel. Towards a standard for embedded core test: An example. In *International Test Conference*, Atlantic city, NJ, September 1999.

[3] M. Benabdenebi, W. Maroufi, and M. Marzouki. Casbus: A scalable and reconfigurable test access mechanism for systems on a chip. In *IEEE Design Automation and Test in Europe (DATE)*, March 2000.

[4] A. Jas, J. Ghosh-Dastidar, and N. A. Touba. Scan vector compression/decompression using statistical coding. In *17th IEEE VLSI Test Symposium*, San Diego, California, April 1999.

[5] A. Jas and N A. Touba. Test vector decompression via cyclical scan chains and its application to testing core-based designs. In *International test conference*, pages 458–464, 1998.