Design for Test of Embedded Microprocessors

Bilal Khan Taherkheli (s010599) Christian Park Kjeldsen (c971469)
Felipe Rivera (s010840) Asger Stenberg (c971431)

Abstract

This document presents a discussion of three topics of design for testing in relation to embedded microprocessors: overall test objectives and design for test strategies, the use of IEEE 1149.1 boundary scan, and the concepts of structurally testable delay paths and functionally testable paths.

1. Introduction

Integration of embedded cores into a larger design presents a special case for typical design for testing methodologies. This report discusses some of the topics present in this case.

The first section of this report discusses the specialization of test objectives and design for test strategies that had to be done for a commercial embedded microprocessor. Section two discusses the use of the IEEE 1149.1 boundary scan standard when designing an embedded core. Finally section three presents a discussion of structurally testable delay paths and functionally testable paths. Conclusions are given at the end covering the three discussed topics.

2. Overall test objectives and design for test strategies

The motivation for implementing design for test strategies may not be obvious at first glance: The added test structures do not improve design functionality nor make the circuit faster or less power consuming. On the contrary, design for test requires additional hardware causing performance degradation as well as complexity increase.

As with all other design efforts the driving force behind introducing design for test is the assumption that this will reduce the final cost of the integrated circuit by easing the task of testing the circuit. Depending on the design complexity, volume, and application demands, testing may be responsible for between 30 to 80% of chip costs [1], thus when trying to cut down production expenses this is an obvious area to look for potential savings. Other reasons for adding more test structures may be demands for reliable and safe systems to use in critical and highly dependable applications such as in the aerospace or health industry.

The discussion in this section will try to make a general approach to design objectives and strategies for testing as a whole, only going into detail where embedded processor cores differs from the general approach.

Lastly the test objectives and test strategies for the Motorola ColdFire version 3 embedded processor are discussed.

2.1. Objectives of Testing

Testing a design for errors and flaws is not at all a straightforward task. Defining this task can be done by setting a series of objectives to guide test development. Two major groups of test objectives can be formed: these are the functional and the economical ones.

2.1.1 Functionally Related Objectives

Functional errors in a circuit can be caused by either the designers of the circuit making an error or it can be introduced during production and appear as a random error somewhere within an integrated circuit. Design errors present before production are not at all uncommon and may be very hard to test for. However, design integrity is not the subject of this text and thus will not be discussed in further detail.

Obvious objectives of post-production testing are the wish to verify functional integrity and performance requirements of an integrated circuit. More objectives in this category may be defined by external requirements such as minimum stuck-at fault detection. Specific requirements like this may be dictated by specifications or set by marketing.

2.1.2 Cost Related Objectives

All companies and their products are driven by the requirement to be economically profitable. Profit in the IC companies is generated when their ICs are bought and in a tough market the pricing of a chip may well be the only factor tipping the favor over competitors, even when they offer chips with higher functionality. Therefore the economic feasibility of IC design is a very important factor that should not be ignored.

As a result of design for test, the price per integrated circuit can be increased by three major factors

- *Area impact* Silicon area increase from special test-structures will reduce total number of chips per die and lower yield, thus increasing the price per chip.
- *Testing time* When tests are applied to the newly produced IC the time to test each chip is crucial. If testing time is too long more test equipment is needed to keep up with production pace thus increasing the initial costs of setting up a production. This is a one-time expense and it can be reduced by accepting lower production throughput, though this is rarely an option.
- *Degree of testability* If a design is hard to test the test engineers may spend a lot of resources developing complex test programs to achieve a required fault coverage. In turn these tests may require expensive test equipment adding to production costs, but these are one-time expenses.

In the ColdFire project a test area impact of 5% was defined and the time to test a circuit was required to be maximum 5 seconds [2]. These numbers were published in 2000 and one would expect at least the test-time (of a similar sized circuit) to have decreased in the last couple of years. The degree of testability was improved with the introduction of test techniques such as memory BIST.

2.1.3 Test objectives for the ColdFire microprocessor

The targeted objectives (goals) defined for the microprocessor are

- *Low cost*
- *Consistent methodology*
- *Automation*
- *Accountability*

One of the reasons behind Motorola's emphasis on low cost testing is to make life easier for the clients. If Motorola manages to conceive a fast, simple and efficient test, it becomes a solid selling argument. Thus one would expect that a low complexity objective would be given a high priority in test strategy development.

A consistent methodology will allow to reuse it for future designs.

Automation is geared towards the reuse of already existing tools and processes for testing, like ATPG. Finally it is very important to be able to measure the effectiveness and usefulness of testing to have feedback to improve the testing process and to have figures that the clients can use to make their purchase decisions.

2.1.4 Wrap Up

The two groups of objectives above are considered independent, however, in reality they can hardly be completely separated. For instance a fault coverage functional requirement is also an economic one, since covering a small amount of errors will lead to a high number of defective circuits that will be expensive to replace.

Objectives may be prioritized depending on the design at hand. For instance in a high volume production, the reoccurring expenses (e.g. the die area impact) should be kept low while the one-time expenses (e.g. tester equipment) should have a big tolerance. The opposite may be true for productions targeting smaller market segments.

Consequences of not fulfilling the objectives within the functional may not be clear, while failing economic objectives is indeed very visible. Still it is important to take all objectives into consideration when developing a test strategy for a given design. Nothing is black and white and objectives may have to be prioritized over others depending on the design at hand.

2.2. Design for test strategies

Once the objectives of testing have been investigated and adapted for a given project, a test strategy for this project can be developed so that it fulfills the previously defined objectives.

A test strategy defines a set of overall rules and tools that must be implemented and used in the process of designing an integrated circuit as well as in the subsequent testing. For example it is a test strategy to decide on implementing memory BIST (Built-In Self Test) based on the need of reducing post-production testing time, thus lowering costs.

The following sections comment on the test strategies for an embedded microprocessor core, Motorola ColdFire (versions 2 and 3) described in references [2] and [3].

2.2.1 The ColdFire Test Strategy

This section is meant as a primer on how a test strategy may look and what considerations could be taken into account. Because the available literature does not go into depth about the background and internal priorities and decisions made by Motorola during test strategy development, the authors of this report will also try to discuss on some of the not so obvious conclusions that the literature presents.

Many testing strategies for the ColdFire version 3 were direrctly taken from those defined for the version 2. But some additions had to be made to the design as part of new strategies and some of the existing ones being refined.

The test methodology is driven mainly by the limititation in the Automated Test Equipment (ATE), the pad frequency capabilities and the existing ATPG tools. The company saves money by using existing ATE and by reusing existing ATPG tools. Also this saves development time because the test engineers work with equipment and tools they are familiar with.

New functionality introduced in version 3 of the processor required special attention to comply with testing objectives.

Next comes a list of existing testing techniques being used already for the ColdFire version 2

- Path delay testing
- At speed scan
- Iddq
- scan-based burn-in
- DMA-based memory testing
- Sequential testing

DMA-based memory testing was dropped in favor of MemoryBIST. The reasons are discussed further in this section. Sequential ATPG was completely eliminated as well, freeing memory in the ATE and saving time.

The new strategies that had to be developed for the ColdFire version 3 were related to three new characteristics of the design

- Multiple clock domain frequencies
- The internal PLL source
- The doubly embedded nature of the core memory arrays

The multiple clock domain problems are not directly related to the core, but to the use of slow peripherals around the core, hence it won't be discussed although it created its share of problems and was one of the challenges faced by the test engineers. Thus, potential clients may want to have a single clock domain.

Another big challenge derived from the characteristics mentioned above was the testing of the memories.

DMA-based memory testing done previously for ColdFire version 2 presents several disadvantages: it requires a big number of signals going in and out of the core impacting routing requirements; is very design specific; it makes complete fault coverage of the SRAM impossible, and path delay couldn't be performed completely. Therefore a new approach to test the memories was selected: MemoryBIST. MemoryBIST requires a relatively big number of gates (4% of the silicon in the particular case of the ColdFire version 3). Still the advantages provided by it more than compensate for this area expense. One of these advantages is not exposing any internal signals outside the core for testing purposes, which translates into a more modular, nicely wrapped core.

As we have discussed, time is money. So another contributor to testing time was decreased: the pauses for retention tests for the three embedded memory arrays were decreased from six to only two.

Finally, although MemoryBIST is a good thing it solely covers the memory cells. Memory inputs and outputs still have to be tested, thus controlabillity and observability registers to allow scan testing of the adjacent logic to memories were added. The result was an increased fault coverage. This technique was named Flow-Through RAMs for Scan Control/Observe

In a nutshell, costs were within constraints, a consistent methodology was followed, automation was attained and metrics were given (see Table 1), thus fulfilling the test objectives

Type	Pattern count	Clock cycles	Fault coverage (%)
Stuck-At	2,098	356,660	97.06
I_{DD}	1	340	No additional
I_{DDQ}	20	3,400	(83.22 I_{DDQ} coverage)
Transition delay	3,137	533,290	79 transition/ no additional stuck-at
Path delay	4,000	680,000	0.15
JTAG	—	15,846	1.71
Memory data retention	—	95	No additional
PLL tests	—	14,518 (61 vectors)	0.36
Miscellaneous	—	3,154	0.44
Total		99.73	

Figure 1. Coverage statistics for the MCF5307

2.2.2 Wrap Up

It seems to be a trrend that the IP-Core design companies move most of the testing into the core. Thus removing the need of complex ATEs. This paper shows that trend. This will allow the buyer of the IP-Core to focus its testing efforts towards their own design knowing that the IP-Core demands from the testing strategies will be small.

Whether the previous scenario becomes a reality, will depend only if the clients are willing to sacrifice speed and area. With the time-to-market requirements on today's electronic industry and the ongoing miniaturization of the basic transistor, which allow very high clock rates, this may well happen in a not so distant future. It will be interesting to see if the IP-Core developers can deliver. and this paper shows us that they are going in this direction.

3. Use of IEEE 1149.1 boundary scan design

The discussion of this question is mainly based on a set of articles found at DTV [4],[5],[6],[7],[8],[9],[10],[11],[12] and the two given articles [2] and [13]. We try to reveal the purpose of using the IEEE 1149.1 boundary scan standard (JTAG) in connection with embedded cores, why some people use it, and some of the aspects for the future. For the functionality of the JTAG standard the reader is referred to [14] since it will not be discussed here. The discussion can therefore be seen as an overview of the work done regarding this subject.

3.1. Purpose of the IEEE 1149.1 with embedded cores

The JTAG standard does not only define a way to test the I/O-pins of an IC, but can also be used as an interface to facilitate other kinds of tests, access internal scan registers, or even programming an FPGA. Thus the JTAG is very versatile and has therefore been implemented in many stand-alone ICs today.

However, since embedded cores are meant for modular ICs or System-on-Chip (SoC), the use of the JTAG boundary scan test is not directly applicable. The main reason for this is that the standard has not been designed for a hierarchy of test access ports (TAPs), i.e. some TAPs at core level and one at chip level [4]. Of course a JTAG boundary scan test can be performed on the complete IC, or be an interface to a test of the individual cores, which is commonly used [2]. However, the principles behind the JTAG boundary scan are good and can be extended to individual embedded cores.

One of the main ideas of the boundary scan is to test the interconnects between two different chips (i.e. a test of connections made in different domains - wires on chip and wires on the PCB). For a SoC or a modular IC the interconnects between the different embedded cores will usually be made in the same way as the embedded core itself. Hence the test of the interconnects will more often happen in a functional test. The JTAG scan chain will instead be used to test whether a certain state internal of the core is reached.

In [9] a work-around of the problem with hierarchical TAPs is made. This is done for the G2 PowerPC^{TM} 603e^{TM} embedded microprocessor core. It is based on the work done in [4]. Here the TAP of each embedded core is provided with a select and enable signal. They are controlled from a TAP linking module (TLM) placed on the chip (see Figure 2). These additional signals will however alter the FSM of the TAP. Besides, this work-around requires the same modification done for all the TAPs on the chip - thus deviating somewhat from the concept of a standard.

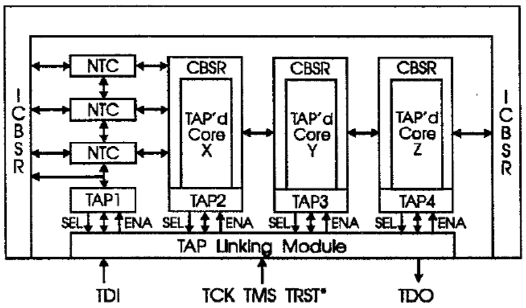

Figure 2. A TLM is seen to control the different TAPs on chip. (BSR: boundary scan register, NTC: non-TAP'ed cores)

Another approach is described in [5]. Here a hierarchy is made, where the lower level TAPs are connected serially, as a part of the higher level TAP (see Figure 3). Hence the highest level TAP does not deviate from the standard, but it requires some knowledge of the hierarchy to be useful when performing the test.

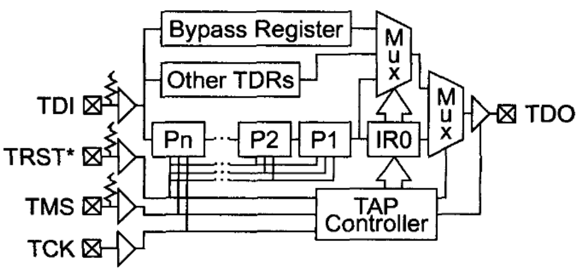

Figure 3. A chip level TAP made by a serial connection of the embedded TAPs, the bypass registers, and test data registers. The Ps are the embedded TAPs

At the moment a working group is developing the IEEE

P1500 standard, which should define the test of embedded cores. But this will be discussed in more detail later.

One minor detail that is easily over-seen is the question: What if the TAP controller doesn't work. This is the question dealt with in [7]. Here a concurrent error detection (CED) approach is used, which reduces the complexity of testing the TAP controller. Besides, a very short test pattern is shown to be enough for testing this design.

3.2. Why use IEEE 1149.1?

The JTAG standard is a relative old standard (1990) which has proven its usefulness regarding chip level design through the wide acceptance and implementation of it. Besides, it has been well documented. The major strengths of the standard is the low number of pins it requires, the simplicity of its design, the versatility previous mentioned, and the ease of implementing particular functionalities without losing any compliance with the standard.

When designing a PCB it has been the trend to stop using chips made in-house and to start buying IP-Cores from small, specialized companies. Hence a standard interface is necessary to perform testing. Here the IEEE 1149.1 has proven its worth. In [2], [10] and [11] it is described how the standard has been implemented respectively in the Motorola third-version ColdFire, the AMD-K7™ and the Sun UltraSPARC™-III microprocessors. However these designs have been implemented by taking the microprocessor core and adding the TAP controller next to it on-chip.

In [2] and [11] it is mentioned that the DFT strategies used are more or less standardized with the rest of the microprocessor families. This might be one of the main reasons why not all companies (or rather design groups) implement the JTAG boundary scan design even if it is feasible: It takes time to implement a new thing. Often it is faster just to do what you are used to, resulting in a better time-to-market factor. However, the JTAG standard is so well described that it should be possible to implement it relatively fast.

Another reason for not using the JTAG module could be company policy: Companies might have their own way of testing and they are happy with it. But, as mentioned earlier, with the greater demand of compatibility among IP-Cores from different companies, the time and money that would be used implementing it would be well spent.

The pros and cons concerning the standard mentioned here in connection with chip design can be reused when the focus is shifted towards embedded cores for modular ICs or SoC. The main question is: What is this standard going to be? Should it be something very close to the IEEE 1149.1, or should it be a whole new standard?

In [9] they've taken an already designed stand-alone microprocessor, the Motorola PowerPC™ 603e™, and tried to make it into an embedded core. Again in such a redesign time-to-market is a very important factor. Thus a fundamental redesign is not a solution. As they already had a TAP in the design of the stand-alone IC, a chip designer using the embedded core would necessarily run into the problem of having more JTAG modules on one chip. Simply to remove the TAP controller was not an option, since the coupling of clocking and debugging features were made here. They solved the problem by adding the select and enable signals from [4], which only required minor modifications to the design. However it is not standard-compliant, hence the future will probably be different.

3.3. The future of the IEEE 1149.1 with embedded cores

As already shown the JTAG standard is not suitable for implementation into embedded cores. As the market evolves, chips will probably end up being made from a set of black-box embedded cores designed in other, smaller companies. Hence some standards have to be implemented to ensure compatibility between the embedded cores.

What needs to be standardized and implemented is the following (see Figure 4)

- A way to switch between core-internal, core-external and functional modes, which can be implemented in a hierarchical way. This is referred to as a wrapper.
- A way to access this wrapper: a Test Access Mechanism (TAM).

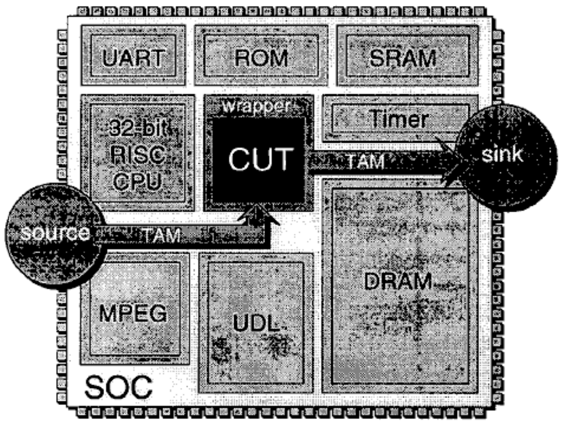

Figure 4. The wrapper and the TAM is seen.

These two items form the test access infrastructure of the chip. They are both part of the IEEE P1500 standard which is under development. However, other companies, e.g. Philips, have already created their own wrapper. In [8] a solution to how to combine these two wrappers is given.

The IEEE P1500 is based on some of the same principles as the IEEE 1149.1, but many differences appear.

Not all people agree on the approach of the IEEE P1500 working group. In [6] a rather critical stand is taken. The main reason is that the IEEE P1500 is not integrable with a chip level JTAG module, without violating the IEEE 1149.1 standard. Besides the support of a hierarchy is very inadequate. An approach as the one used in [5] would be a lot better here.

Another question that arises is how easy it is to integrate the standard into a program for automated test generation. In [12] a method is described on how to formalize and implement such a program, which is very well-suited for SoC's (but in general just modular testing of large ICs).

One of the advantages of the IEEE P1500 is that it is easy to model, hence it is easy to implement into a program for automated test generation. An approach as the one presented in [12] might be the way to do it. The approach used in [5] for designing the test access infrastructure of the chip is, however, not as easy to model. This might be the reason for the IEEE P1500 work group not to consider this approach in more detail. However, the author of [6] claims that the P1500 work group has given no rational reason for not considering such an approach.

3.4. Wrap up

It is obvious that the IEEE 1149.1 is of limited use when designing SoC's or modular ICs. As we are just students taking a course on IC testing it will not be possible for us to say what will be the future. However, at least two positions can be seen: Those who wait for the IEEE P1500 to be widely accepted and those who prefer a more IEEE 1149.1 based standard (some of the work has been done by the 1149.2 work group - where hierarchy issue has been addressed).

4. Structurally testable delay paths and functionally testable paths

In modern deep sub-micron technologies small processing defects can cause unacceptable propagation delays and if such a defective gate lies on the critical path, this can result in a circuit failure.

To ensure that a circuit meets the timing specification and behaves correctly when operated at full speed, delay path testing must be employed. Two fault models are commonly used for verifying temporal correctness of manufactured circuits: path delay fault model and gate delay fault model.

The path delay fault model assumes the propagation delay of a target path exceeds the clock period while the gate delay fault model assumes the propagation delay of a gate/node exceeds its specified limit. The pros and cons of both models have been discussed in many articles. Path delay fault model is the preferable choice as it models delay faults more closely as discussed in [15].

4.1. Structurally Testable Delay Paths and Functional Testing

A path is structurally testable if there exists a test vector pair applied through the enhanced or standard full scan chain, which sensitize the path.

The details about the application of scan design are too complex to be described here. A rapid advance in semiconductor technology has resulted in more and more transistors on a chip, and higher and higher operating rates are being achieved. This has affected delay testing of microprocessors because testers cannot keep pace with such high on-chip clock speeds. The trend has been to employ built-in self-test (BIST) techniques.

BIST techniques offer the availability to apply and analyze at-speed test signals on chip with greater accuracy than that of the tester. Structural BIST techniques like scan based BIST offer good test quality but at the expense of increased chip area, reduced performance, and increased design time overhead.

Another major problem with structural BIST techniques is that due to the application of nonfunctional, high-switching random patterns, they consume more power than what will be consumed under normal operating conditions.

These problems can be overcome using the instruction set of on-chip processors as the test vectors. This is called functional test. Assuming any arbitrary vector pair can be applied to a path, it is structurally testable if a vector pair can detect a fault given a sensitization requirement. This implies that only an enhanced full scan with each flip-flop having the ability to store two bits of data can achieve this. With standard full scan not all vector pairs can be applied to the combinational part of the circuit. Therefore, not all structurally testable paths are testable under standard full scan.

In functional testing, where instruction sequences are used as the test vectors, certain temporal and spatial correlations are imposed among registers/flip-flops and thus among inputs/outputs of the embedded block of the processor. This results in a significant decrease in the size of the set of vectors applicable for delay testing.

Not all of these reduced set of vector pairs are significant for delay testing because in certain situations when a certain vector pair is applied some of the registers are not enabled by the controller in the immediately following clock cycle. The delay fault effects propagated to them are lost as they are not latched. Therefore, it is very much possible that some paths are structurally testable but not functionally testable.

It can be concluded that delay faults fall into three categories

1. Enhanced-scan structurally testable

2. Full-scan structurally testable

3. Functionally testable. Functionally testable paths are also full-scan structurally testable; full-scan structurally testable paths are also enhanced-scan structurally testable as shown in Figure 5.

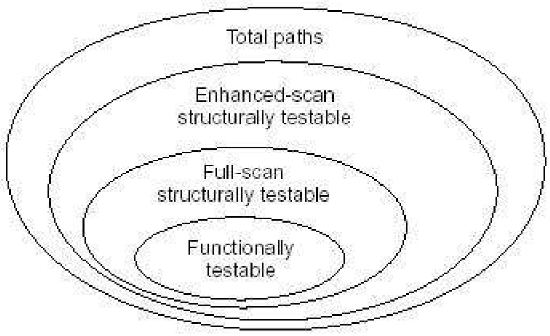

Figure 5. Classification of a path delay fault

4.2. Functionally Testable Paths

A path is functionally testable if there exists a functional test for the path, otherwise it's a functionally untestable path. The delay defects on any functionally untestable path would not cause chip failure unless the delay exceeds twice the clock period, in which case they are likely to be caught by tests for transition faults using stuck-at fault models. Thus, delay testing need not target functionally untestable faults. This means that if we can identify functionally untestable paths in a microprocessor, we can gain some significant advantages which are discussed later in this text.

Consider the data path of Parwan as shown in Figure 6, containing an ALU, an instruction register (IR) and an accumulator (AC). Data inputs A7-A0 and B7-B0 are connected to the internal data bus and AC, respectively. ALU selects appropriate function to perform through the control inputs S0-S2. The paths that start from A7-A0, B7-B0, and S2-S0 and finish at the inputs of AC and IR are structurally testable under enhanced scan if we can find a vector pair to test them.

If, however, we restrict our vector pairs to be derived from instruction sequences, some of the paths may not be functionally testable even when they are structurally

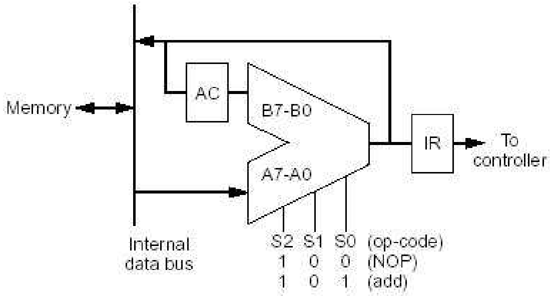

Figure 6. The data path of Parwan

testable. This again emphasizes that functionally testable delay paths are a smaller subset of the structurally testable delay paths.

For example in the case of Parwan it can be shown that whenever there is a falling transition on S1, AC and IR can never be enabled in the immediately following clock cycle. Therefore, even though a falling transition can be generated on S1 and propagated to the ALU output, no register will be able to latch the transition. The paths starting at S1 and ending at the inputs of IR and AC are functionally untestable even though they might be structurally testable.

Two classification algorithms, one for the data path and another for the controller of the processor, targeting functionally testable faults are described in [13]. The data path and the controller are considered separately because of their different natures. Data path activities depend heavily on the controller while data path outputs do not influence the controller that much. Moreover, data path can be partitioned into a set of combinational blocks and registers, while partitioning the controller is complicated.

The experimental results for two processors, Parwan and DLX are shown in Tables 1 and 2. As can be seen in the tables, the number of functionally testable paths is significantly smaller than the structurally testable paths. The differences between structurally testable paths and functionally testable paths are relatively higher for data paths, because of its partitioning, than for the controllers as discussed in [13].

		Parwan		DLX	
		NR	**FS**	**NR**	**FS**
Total paths		29,542	29,542	429,838	429,838
Structurally testable (enhanced scan)	Paths (%)	89.6	91.9	90	97
	CPU (sec)	17	11	2,042	1,183
Functionally testable (using instruction sequences)	Paths (%)	14.2	16	20	39.2
	CPU (sec)	15	13	7,622	8,494

Table 1. Results for data paths

		Parwan		DLX	
		NR	FS	NR	FS
Total segments	2,952	2,952	17,708	17,708	
Structurally testable (enhanced scan)	Paths (%)	57.2	66.2	35.4	50.9
	CPU (sec)	4	5	76	97
Functionally testable (using instruction sequences)	Paths (%)	53.8	63.5	33	45.7
	CPU (sec)	7	8	157	184

Table 2. Results for controllers

4.3. Applications

The specification of functionally untestable paths has some important applications. These are

1. Realistic fault coverage measure
2. Reduced test generation effort
3. Appropriate path selection in delay testing
4. Identification of true critical paths
5. Help in synthesis process

Thus BIST must target functionally testable paths as these are crucial to circuit performance. Classification of functionally untestable paths helps in reducing the BIST circuitry as it guides whether or not to insert BIST circuitry at some place in the circuitry.

Static timing analysis of a circuit to identify a critical path and then spending a lot of time on improving the circuit performance is not a very wise idea because not all the critical paths are necessarily testable. Functionally untestable paths should not be considered as a critical path and no optimization efforts should be wasted on them as they do not have any affect on circuit performance. In complex circuits like microprocessors, only a subset of paths is tested due to the very large number of paths.

Path selection algorithms affect subsequent test generation and final test quality. The path selection algorithms should only select functionally testable paths. Another area in which specification of path delay faults is desirable is the logic synthesis quality and efficiency of timing optimization process. If the synthesis process uses only functionally testable paths for optimizations, it ensures an efficient usage of its limited resources as a consequence of which the circuit is optimized for better timing and performance.

5. Conclusions

The goals of this report were to investigate design for test topics for embedded processor cores. Specifically three topics were investigated: test objectives and design for test strategies, use of IEEE 1149.1 boundary scan in IP-Cores, and structurally testable paths and functional testing.

Based on the ColdFire articles it was possible to make a general overview of test objectives and design for test strategy. Also it was seen that prioritizing objectives is possible, and that this will affect the test strategy and cost.

Also investigating the test strategy of the ColdFire processor has given an insight into the diverse requirements and pitfalls of testing when working on complete embedded cores and a small glimpse of what we can expect in the future.

By investigating the use of IEEE 1149.1 in several commercially available devices and how the standard is defined, it was found that the current boundary scan standard had limited use for systems-on-chip featuring embedded cores. However several IEEE activities have been started, which address the issue of testing of embedded cores.

The difference between structurally and functionally testable delay paths were discussed. It was shown that functionally testable paths is a subset of structurally testable paths. When the issue of delay test is concerned only functionally testable paths should be considered. Surely the goals have been fulfilled and this report provides and insight into design for test through the help of test objectives and strategies, boundary scan, as well as information on functionally and structurally testable paths.

References

[1] A. R. et al, “Design for test strategies for analogue and mixed-signal integrated circuits,” *38th Midwest Symposium on Circuits and Systems. Proceedings*, vol. 2, pp. 1139–44. Cat. No. 95CH35853. ISBN 0780329724.

[2] A. C. et al, “Test development for a third-version ColdFire microprocessor,” *IEEE Design & Test of Computers*, pp. 29–37, Oct-Dec 2000.

[3] D. A. et al, “A case study of the test development for the 2nd generation ColdFire microprocessors,” pp. 424–432, International Test Conference, 1997.

[4] L. Whetsel, “An IEEE 1149.1 based test access architecture for ICs with embedded cores,” pp. 69–78, IEEE International Test Conference, 1997.

[5] S. F. Oakland, “Considerations for implementing IEEE 1149.1 on system-on-a-chip integrated circuits,” pp. 628–637, IEEE International Test Conference, 2000.

[6] B. I. Dervisoglu, “A unified DFT achitecture for use with IEEE 1149.1 and VSIA/IEEE P1500 compliant

test access controllers," pp. 53–58, DAC, June 18-22 2001.

[7] S. Mitra, "Design for testability and testing of IEEE 1149.1 TAP controller," Proceedings of the 20^{th} IEEE VLSI Test Symposium (VTS'02), June 18-22 2002.

[8] E. J. M. et al., "Wrapper design for embedded core test," pp. 911–920, IEEE International Test Conference, 2000.

[9] C. H. et al., "Design and implementation of the G2 PowerPC™ 603e™ embedded microprocessor core," pp. 473–479, IEEE International Test Conference, 1998.

[10] T. J. Wood, "The test and debug features of the AMD-K7™ microprocessor," pp. 130–136, IEEE International Test Conference, 1999.

[11] F. Golshan, "Test and on-line debug capabilities of IEEE std 1149.1 in UltraSPARC™-III microprocessor," pp. 141–149, IEEE International Test Conference, 2000.

[12] E. J. Marinissen, "The role of test protocols in automated test generation for embedded-core-based system ICs," *Journal of Electronic Testing: Theory and Applications*, vol. 18, pp. 435–454, 2002. Kluwer Academic Publishers.

[13] W.-C. L. et al., "Functionally testable path delay faults on a microprocessor," *IEEE Design & Test of Computers*, vol. 17/4, pp. 6–14, October 2000.

[14] S. L. Hurst, *VLSI Testing: digital and mixed analogue/digital tech*, vol. 9 of *IEE Circuits, Devices and Systems Series*. The Institution of Electrical Engineers, 1998.

[15] 02208 Test of ICs Team 1, "Delay test," 2002. Report presented as part of the course 02208 Test of ICs at DTU.