ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

A New Approach for Generation of Test Program for Detection of Hardware Fault in VLIW Processor

Mani S[1], Nikil Sathish[2]

PG Scholar, Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India[1]

Asst. Professor, Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India[2]

ABSTRACT: Very long instruction word processors are increasingly employed in a large range of embedded signal processing which is provide high performances with reduced clock rate and power consumption. Software-based self-test (SBST) methods are consolidated and effective solution to detect faults in the processors both at the end of the production phase or operational life. In this report we present a new method for automatic generation of efficient test programs specifically oriented to VLIW processors. The proposed method based on generic SBST algorithms automatically generates effective test programs. It is able to reach the detect fault, while minimizing the test duration and test code size. The method consists of four types of methods and can deal with different VLIW processor models. The main goal of the project is to show that in the case of VLIW processors, it is automatically generate an effective test program able to detect fault with minimal test time and required resources.

KEYWORDS: Software-based self-test (SBST), test program generation, very long instruction word (VLIW) processors**.**

I. INTRODUCTION

Mainly due to the semiconductormanufacturing process and to the increasinglyhigh operation frequency of IC, processor chips face growing testing fields for the problems. Moreover, since the productionprocesses are highly stressed, phenomena like aging of the circuit may increase the occurrenceof permanent faults in the systems, even during the circuitoperational . For these reasons, new test solutions are being investigated in order to provide criticize for adequate coverage with acceptable costs (e.g., in terms of test time, silicon area overhead, and required test infrastructure). A promising approach for processors and processor-based systems (e.g., systems on a chip, or SoCs) corresponds to called software-based self-test (SBST) [1]: the basic idea is to generate test programs to be executed by the processor and able to absolutely exercise the processor itself or other components in the system, and to detect possible faults by looking at the produced results.

One of the main advantages of SBST lies in the fact that it does not require any extra hardware; therefore, the test cost is reduced and any performance or area penalty is avoided.Moreover, the SBST approach allows at-speed testing, and can be easily used even for on-line testing. For these reasons, SBST is increasingly applied for processors and SoC testing, often in combination with other approaches.

Among the various microprocessor architectures, very long instruction word (VLIW) processors were demonstrated to be a viable solution especially for applications demanding high performance while exposing a considerable amount of parallelism, such as several digital signal processing algorithms used in multimedia and communication applications.

VLIW processors are currently adopted in several products, in particular for embedded applications, and the problem of testing them is increasingly relevantVLIW processors are characterized by a pipelined architecture with multiple functional units (FUs). Unlike superscalar processors, VLIW processors do not include any

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

significant control logic, since instruction scheduling is completely performed by the compiler. This implies that the hardware complexity is far lower for superscalar processors, while the compilation steps become more complicated. Consequently, the control hardware of the processor is much more easily testable than in other processors (e.g., the superscalar ones). Another key feature of VLIW processors is the instruction format. In fact, VLIW processors are characterized by grouping several instructions (named microinstructions) into one large macro-instruction (also called bundle), where each micro-instruction within the bundle is executed in parallel distinct computational units, referred to as computational domains (CDs). In VLIW architectures, the scheduling of the operations is fully performed at compile time; the compiler is responsible for allocating the execution of each instruction to a specific FU.

II. VLIW BACKGROUND

A VLIW processor is characterized by the fact that all the operations are executed by parallel CDs, each characterized by its own FUs; the scheduling is completely static, being fully defined at the compile time, as described in Fig. 1.As illustrated in Fig. 2, the assembly code for a VLIW processor is rather different from the point of view of the machine code from that for a superscalar processor: several instructions are classified grouped together in a single macro-instruction (also called Bundle) and each instruction is assigned for execution to a specific CD. Consequently, in a VLIW processor, there is not any hardware instruction scheduler, and the detail typically performed by this component is done by the compiler. In this way, the power consumption is reduced and the silicon area occupied is far less that the area occupied by a traditional superscalar processor, while the design complexity is dramatically reduced.

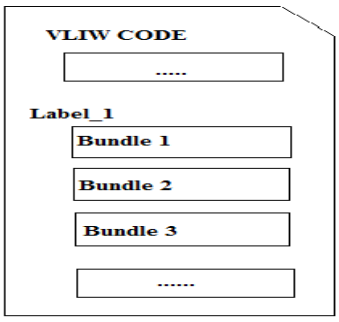

Fig 1 Example of vliw code

Finally, given their high regularity, VLIW architectures can be easily customized to efficiently perform any given application. In fact, a general VLIW processor parametric architecture may have a variable number of CDs and FUs, so that different options, such as the number and type of FUs, the number of multiported registers the width of the memory buses, and the type of different accessible FUs, can be modified to best fit the application requirements. All the features of a VLIW processor are grouped together and listed in the so-called VLIW manifest.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

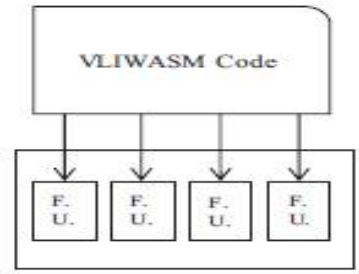

Fig 2 VLIW Architecture

From the software point of view, the machine code is very different with respect to the code of traditional superscalar processors: the VLIW code is composed of a sequence of bundles, each of which contains a number of instructions equal to the number of CDs composing the VLIW processor; each instruction is assigned to a CD, which is responsible for its execution. In Fig. 1, an example of VLIW code is reported, based on the assembly code of a VLIW processor having four distinct CDs [7]. We assume that for each bundle there are four instructions. Considering these features, the VLIW code is more complex than a traditional assembly code, and the size of the code is typically larger. The mapping of instructions to CDs is entirely performed at compile time. It is also up to the compiler to identify the most suitable instructions to be included in each bundle, while guaranteeing that they can be executed in parallel by the different CDs taking into account any possible dependency; if for some reasons there are not enough independent instructions to be assembled into a bundle, the remaining slots are filled with NOP instructions.

III. PROPOSED SYSTEM

In this paper we proposed on the generation of effective SBST test program for the VLIW processors, characterized by maximal fault coverage, minimal duration and size. This method starts from existing functional test algorithm developed for each single FU's type embedded into the processor. Although the characteristic of the fuse used within a VLIW processor are similar to those used in traditional processors, generating optimized code to effectively test these units is not a trivial task. By exploiting the intrinsic parallelism of VLIW processors, it is theoretically possible to reduce the increase in duration and size of the test programs when the VLIW processor size grows.
Experimental data gathered on a case study demonstrate the effectiveness of this approach; results show that this method is able to exploit the intrinsic parallelism of the VLIW processor, taming the growth in size, and duration of the test program when the processor size grows. This method aims at automatically generating the test program for a given VLIW processor starting from the VLIW manifest and from a library of existing SBST programs. The generation of the test program is automatically performed on the basis of the VLIW configuration, and is therefore autonomously tuned depending on the VLIW manifest features. The flow supports some optimizations concerning the execution time, the fault coverage, and the code length, and these optimizations are not correlated to the characteristics of the original test routines.

IV. SYSTEM DESIGN

The execution flow is based on four main steps:

- Fragmentation,
- Customization,
- Selection, and
- Scheduling.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

The flow has three initial inputs, which include two global requirements (the VLIW manifest and the library of generic SBST programs), and a specific input (the SBST program for testing the VLIW register file). The VLIW manifest contains all the features of the processor under test, while the SBST library contains a set ofprogrammable to test the different modules within the processor itself.

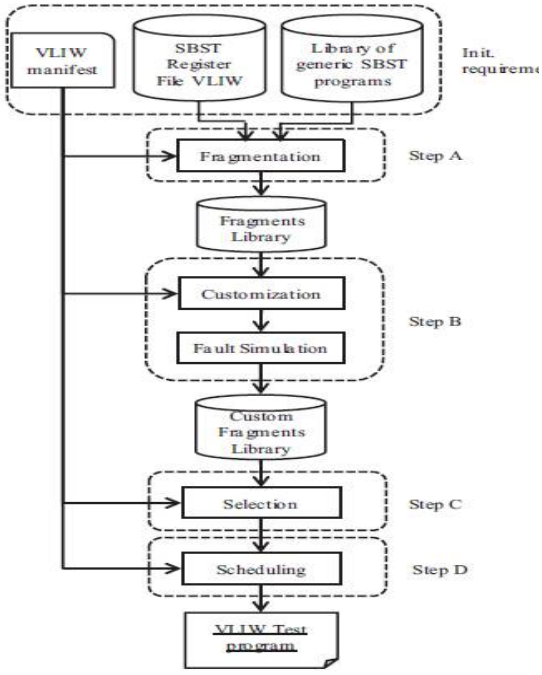

Fig 3 Flow diagram for testing

These two requirements are defined as global since they are configurable depending on the characteristics of the addressed VLIW processor. In particular, the library is a collection of generic SBST programs based on literature test algorithms, it contains the functional test code able to test the most relevant FUs of a generic VLIW processor. The test codes stored into the library are purely functional (i.e., do not require any DfT feature) and are completely independent on any physical implementation of the FU they refer to.
These codes may be based on the techniques used to test the same Fus when used in conventional processors.

4.1 Fragmentation:
The purpose of the fragmentation phase is to minimize the number of test operations in order to create efficient and optimized test programs. The fragmentation phase, illustrated Step A, performs two main tasks. The first is the selection from the library of the test programs needed to test the VLIW processor under test, ignoring those which refer to Fus that are not part of the processor itself. The second task performed by this step is to fragment each selected test program into a set of small pieces of code, called fragments, containing few test operations and the other instructions needed to perform an independent test.

4.2 Customization:
The customization step, illustrated in Step B, is responsible for the translation of the generic architecture independent test programs into the VLIW code, exploiting The ISA of the considered processor. In particular, starting from the fragments library and from the VLIW manifest, the method translates each generic fragment in a custom fragment that can be executed by the processor under test. A custom fragment is defined as a set of instructions belonging to the ISA of the processor under test, which perform several operations in order to test

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

the addressed FU. An example of the customization process, where the code of a Fragment before and after the customization phase is reported.

4.3 Selection of the Custom Fragments:

The selection of the custom fragments, illustrated in Step C, consists in the choice of the test fragments that optimize a set of rules dependent on the requirements desired for the final SBST program. The optimization is performed by the execution of the algorithm described; the algorithm is able to implement two alternative rules. The former aims at selecting the minimum number of custom fragments that allow to reach the maximum fault coverage with respect to all the resources of the processor under test. During this phase, all the fragments are filtered depending on their fault coverage on the full VLIW processor. The filtering

Of the fragments is performed by the execution of multiple algorithm iterations. At each iteration, the algorithm adds to the selected fragment list the fragments that maximize the fault coverage with respect to all the resources of the processor under test. In this way, at the end of the execution, several custom fragments are not selected, since the faults covered by these fragments are already covered by the fragments chosen by the algorithm.If a fault occurs in a functional unit while testing it, than it is not necessary to generate the test codes for that particular functional unit. To do this we use a technique known as “Tiling technique”. This technique is used so the unwanted generation of test code for a faulty unit can be avoided so that the energy and time required to generate the test codes could be saved.

V. EXPERIMENTAL RESULTS

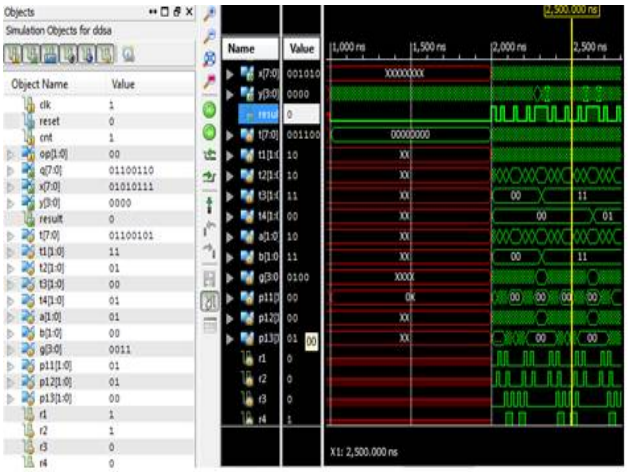

Fig 4:Fault Detection

The figure 4 shows that simulation result of this project. It detect the fault from the VLIW processor based on the equalization of output y and golden output g. The CLK=1, RESET=0, CNT=1 are given as input to the SBST testing. Positive clock edge and reset are OR function, so the register q=01100110, t=01100101. Output y=0000 and the golden output g=0100 the y not equal to g so the result=0 therefore fault detect in the VLIW processor.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 5, Special Issue 2, March 2016

4th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '16)

10th-11th March 2016

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

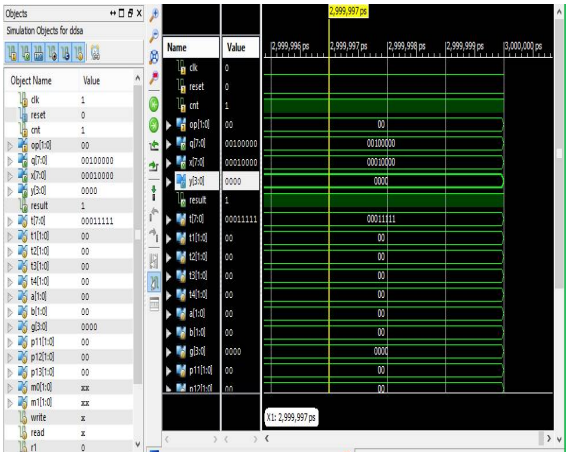

Fig 5:Tilling Technique output

VI. CONCLUSION

The method has been experimentally validated resorting to a representative VLIW processor and generating test programs using a prototypical tool. The obtained results clearly demonstrate to test programs generated by applying generic SBST methods to the specific case of the VLIW processors. More in particular, the method shows that it is possible to develop test programs whose duration and size grows less than linearly with the VLIW parallelism. Then the Tilling technique is used so the unwanted generation of test code for a faulty unit can be avoided so that the energy and time required to generate the test codes could be saved.

REFERENCES

[1] M. Psarakis, D. Gizopoulos, E. Sanchez, and M. Sonza Reorda, "Microprocessor software-based self-testing," IEEE Design Test Comput., vol. 2, no. 3, pp. 4–19, May–Jun. 2010.
[2] J. A. Fisher, P. Faraboschi, and C. Young, Embedded Computing: A Vliw Approach to Architecture, Compilers and Tools. Berkeley, CA, USA: Univ. California Press, Dec. 2004.
[3] M. Beardo, F. Bruschi, F. Ferrandi, and D. Sciuto, "An approach to functional testing of VLIWarchitectures," in Proc. IEEE IEEE Int. High-Level Design Validation Test Workshop, Jul. 2000, pp. 29–33.
[4] D. Sabena, M. Sonza Reorda, and L. Sterpone, "A new SBST algorithm for testing the register file of VLIW processors," in Proc. IEEE Int. Conf. Design, Autom. Test Eur., Mar. 2012, pp. 412–417.
[5] S. Wong, F. Anjam, and F. Nadeem, "Dynamically reconfigurable register file for a softcore VLIW processor," in Proc. IEEE Int. Conf.Design, Autom. Test Eur., Mar. 2010, pp. 962–972.
[6] S. Wong, T. Van As, and G. Brown, "ρ-VEX: A reconfigurable and extensible softcore VLIW processor," in Proc. Int. Conf. ICECE Technol., Dec. 2010, pp. 369–372.
[7] Hewlett-Packard Laboratories. EX Toolchain [Online]. Available: http://www.hpl.hp.com/downloads/vex/
[8] J. Fischer, "Very long instruction work architectures and the ELI-512," IEEE Solid, State Circuits Mag., vol. 1, no. 2, pp. 23–33, Sep. 2009.
[9] J. A. Fischer, "Trace scheduling: A technique for global microcode compaction," IEEE Trans. Comput., vol. 30, no. 7, pp. 478–490, Jul. 1981.
[10] N. Kranitis, A. Paschalis, D. Gizopoulos, and G. Xenoulis, "Softwarebased self-testing of embedded processors," IEEE Trans. Comput., vol. 54, no. 4, pp. 461–475, Apr. 2005.
[11] T. Koal and H. T. Vierhaus, "A software-based self-test and hardware reconfiguration solution for VLIW processors," in Proc. IEEE Symp.Design Diag. Electron. Circuits Syst., Apr. 2010, pp. 40–43.
[12] M. Ulbricht, M. Scholzel, T. Koal, and H. T. Vierhaus, "A new hierarchical built-in self-test with on-chip diagnosis for VLIW processors," in Proc. IEEE Symp. Design Diag. Electron. Circuits Syst., Apr. 2011, pp. 143–146.