An Approach to Direct Memory Access Module Verification

V. Kutsevol <kutsevol_v@mcst.ru>,
A. Meshkov <alex@mcst.ru>,
M. Ryzhov <ryzhov@mcst.ru>,
P. Frolov <opium@mcst.ru>,
ZAO MCST, 24 Vavilova Str., Moscow, 119334, Russian Federation

Abstract. A method of direct memory access subsystem verification used for "Elbrus" series microprocessors has been described. A peripheral controller imitator has been developed in order to provide a flexible way to simulate a wide range of workloads of the direct memory access system without a need for computational overhead caused by simulation of the initialization and operation of the southbridge and its controllers. The imitator has been implemented as synthesizable Verilog module used in verification both with the RTL model and with the FPGA prototype. It can be integrated as a replacement of the I/O link connecting the integrated northbridge with the southbridge thus eliminating the need to simulate extra hardware. This connection method allowed to use a single implementation of the imitator with a complete series of microprocessors compatible with respect to the I/O link interface. The model of the imitator was also included into the functional machine simulator. A pseudorandom test generator for verification of the direct memory access subsystem based on the simulator. The test generator has been developed using library version of the functional machine simulator that allowed to use the simulator as a reference model during the test generation. The consistency of the programming interface of the imitator provides ability to execute generated tests unmodified on the functional machine simulator, the RTL model, the FPGA prototype and even the fabricated microprocessors when integrated in the FPGA I/O link controller. Employment of this method allowed to find a significant number of bugs in "Elbrus" series microprocessors being developed.

Keywords: system verification, functional model, direct memory access, pseudorandom test generation.

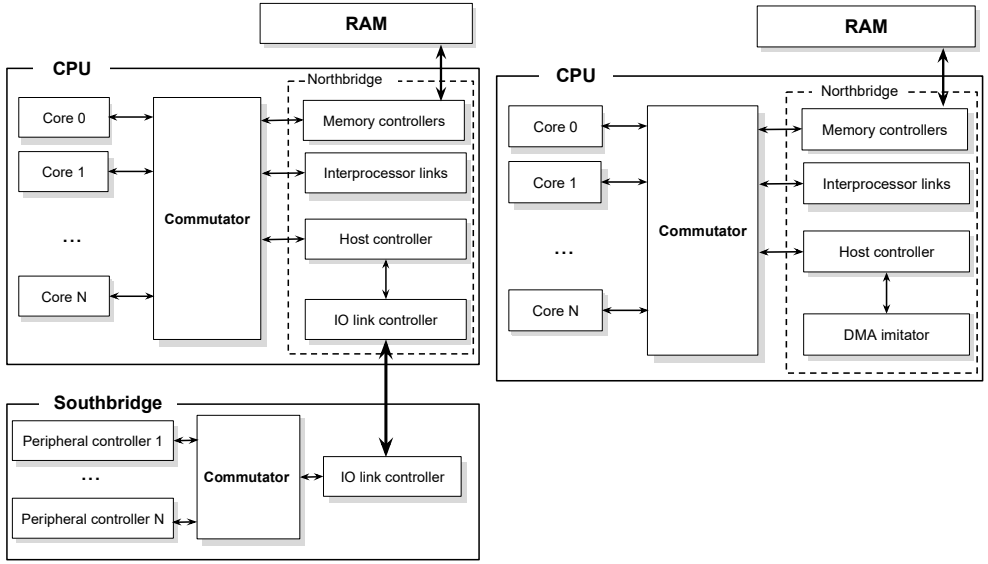
DOI: 10.15514/ISPRAS-2015-27(3)-10

For citation: Kutsevol V., Meshkov A., Ryzhov M., Frolov P. An Approach to Direct Memory Access Module Verification. Trudy ISP RAN/Proc. ISP RAS, vol. 27, issue 3, 2015, pp. 139-148. DOI: 10.15514/ISPRAS-2015-27(3)-10.

1. Introduction

Modern computer systems require very intensive data exchange between the peripheral devices and the random-access memory. In the most cases this exchange is performed by the direct memory access (DMA) subsystem. The increasing demands for the performance of the subsystem lead to an increase in its complexity, therefore requiring development of effective approaches to DMA subsystem verification [1,2].

This article is based on a result of a comprehensive project than combined implementation of a three co-designed verification techniques based on the consecutive investigation of the DMA subsystem employing one the three models: 1) a functional model written in C++ that corresponds to behavior of the subsystem in the environment determined by a real computer system configuration, 2) RTL model in Verilog and 3) FPGA-based prototype. This article describes the first method that enables verifying correctness of the design at an early stage of the verification and eliminate a large quantity of bugs using simple tests.



a). b).

Figure 1. The structure of the computer systems:

a). Real configuration.

b). Model configuration (integration of the DMA imitator into the northbridge).

The most important problem that significantly affects the quality of the subsystem verification is the exhaustiveness of the representation of the external devices connected to it and input vectors they generate. In this case, the problem has been

solved by introducing a device imitating a peripheral controller and capable of generating a comprehensive range of DMA subsystem interaction patterns into the functional model. The basic aspects of DMA imitator implementation are presented in the second section.

The exhaustiveness of the subsystem in question verification is achieved with a test generator allowing to provide necessary inputs using the imitator. The generator produces a test program that performs the DMA imitator scenarios setup for all of its agents, launches their concurrent execution, provides memory access by the CPU cores during the DMA access scenarios execution and checks the final memory state. The generator operation principles are described in the fourth section of the paper.

The generation of final memory state checking code requires a golden model of the memory subsystem being available for the generator. A functional model library that will be described in the third section has been reused from previous projects in order to fulfill this requirement.

2. Peripheral device imitator

Considering the computer system containing the subsystem (fig.1a) in question it should be noted that difficulties connected to precise modeling of the south bridge devices caused by the usage of the complex device drivers can be avoided via imitating behavior of the real DMA agents. A masked DMA copy operation has been used as a basic operation that allows to implement the significant number of the direct memory access scenarios. In order to achieve a high-speed test execution, the imitator is integrated into the IO link between the northbridge and the chipset (south bridge, fig.1b). The positioning of the imitator as a standard IO controller allowed to apply this scheme to any modern Elbrus series processor.

The imitator represents a simplified version of the southbridge. It includes adjustable number of identical agents (fig.2), each capable of working in normal or table modes. In the table mode the memory access scenario specification is simplified by providing them via tables placed in the memory.

Agent is capable of the following operations:

- copying data from one area of the memory to another in normal and table modes,
- reading copy operation parameters from memory,
- data transformation.

The imitator is implemented as a PCI-compatible device, each agent is an independent device that is controlled by a common bus via load and store operations to the configuration space. Agents can perform an exchange with the memory using standard read and write packets. The commutation between the agents is performed by the DMA Switch module.

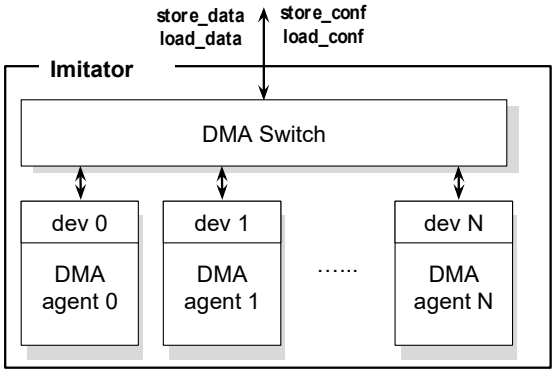


Figure 2. The structure of the DMA-imitator.

The structure of the DMA-agent is shown at this fig.3. `ConfigResigters` module is an array of configuration space registers containing setup operation modes, base addresses and other parameters. In the normal mode the addresses are written to the `ConfigRegisters` are used to access the memory. In the table mode the `TMHandler` module uses written address to fetch and process the table with address of reads and writes. The `Format` module is responsible for masking the data and correct merging of data in the table mode. The `DMAEngine` module is implemented as a FIFO buffer that performs loads and stores of the data using the DMA write and DMA read functions provided by the functional model.

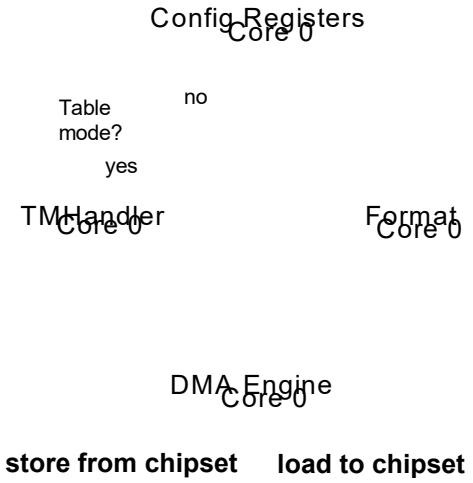


Figure 3. The DMA-agent.

3. Functional model of the DMA imitator

The approach to the problem is based on presenting the direct memory access as two independent modules: the simulator, that imitates the work the computer system architecture objects that are directly employed in the process, and a test generator that provides the modes and parameters for the direct memory access, sets up the logic of the these objects and controls the correctness of the outcome (fig.4). The structural and functional independence of these modules significantly increases the flexibility of the system in such aspects as content and interaction of objects under study, the spectrum of generated inputs and results checking.

The configuration of the simulator that has been developed contains four processor each one containing several general-purpose cores and a northbridge, the southbridge and an imitator that consists of an array of peripheral devices and their interfaces [3]. According to the second section the communications of the imitator and the north bridge are performed by the functions of the programming model described in the PCI standard.

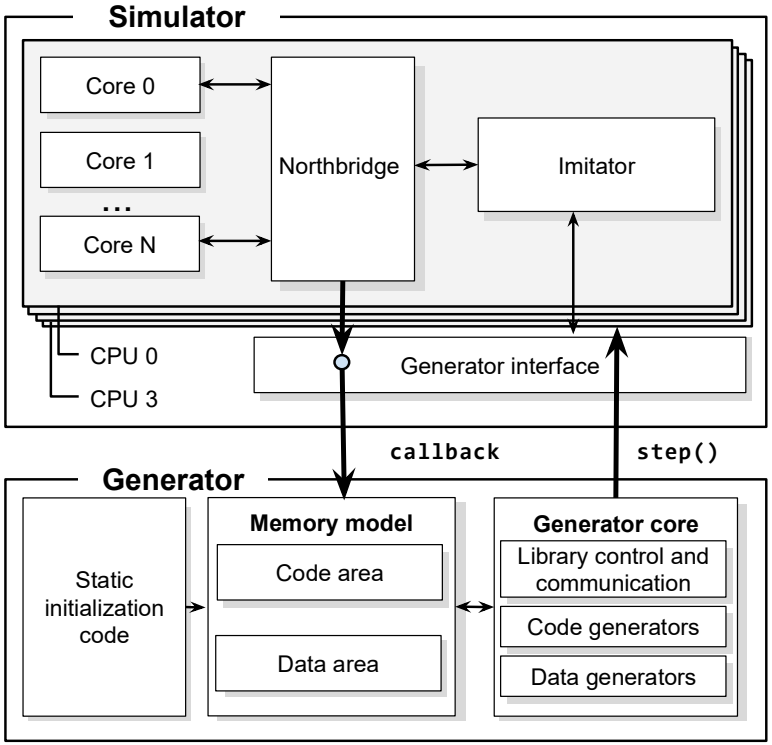


Figure 4. Components of the DMA subsystem functional model.

The simulator works according to interpretation principle [4]. In each virtual tick execution of one command in each of the processor cores is performed. In addition, different asynchronous actions in respect to the commands execution actions such as counter and timer ticks and external interrupt handling are also performed during a single tick.

In order to enable the communication of the simulator with the generator it has been decided to implement a working cycle of the simulator available through a set of library functions.

4. Test generator

The generator contains the static initialization code, the memory model and the core of the generator. The initialization code is a sequence of instructions that performs the initial setup of the hardware performed by the test.

The core of the generator contains the library control and communication module as well as the code and data generators [5]. The library control and communication module is responsible for interaction with the simulator. It invokes the `step()` function that implements execution of instructions of the modeled hardware and the analysis the result of its execution. The code generator writes the code that controls the operation of each of the DMA-agents and the data generator writes the blocks of the data to be send. The flexibility of the DMA-imitator parameterization is fully supported by the pseudorandom test generator that sets up pseudorandom parameters for the DMA-exchange such as addresses of the memory buffers, ranges of the DMA-packet sizes as well as different transfer modes.

Both static initialization code and dynamically generated code is placed into the code area that is one of the components of the memory model. When code fetch takes place during the program execution the requests are directed by the callback function to the code area of the generator. The data area that is another memory model component is handled in a similar manner. The requests for the data --- the loads and stores can be initiated by both the CPU cores and the DMA-agents. All of the requests are redirected to the data structure containing the array dynamically allocated by the data generator.

The step-by-step algorithm of the simulator main modules interaction with the generator is presented in the fig.5.

The general scenario of working with the DMA-imitators has the following outline: the basic system initialization, the initialization of the DMA buffers with the data designated for transmission, the configuration of the DMA-imitator and the launch of the DMA-exchange. Such system parameters as number of processors and available physical address ranges can be varied in a random way to create different DMA routing scenarios. The system initialization procedure can also turn on input/output memory management unit (IOMMU) and fill translation table with random entries.

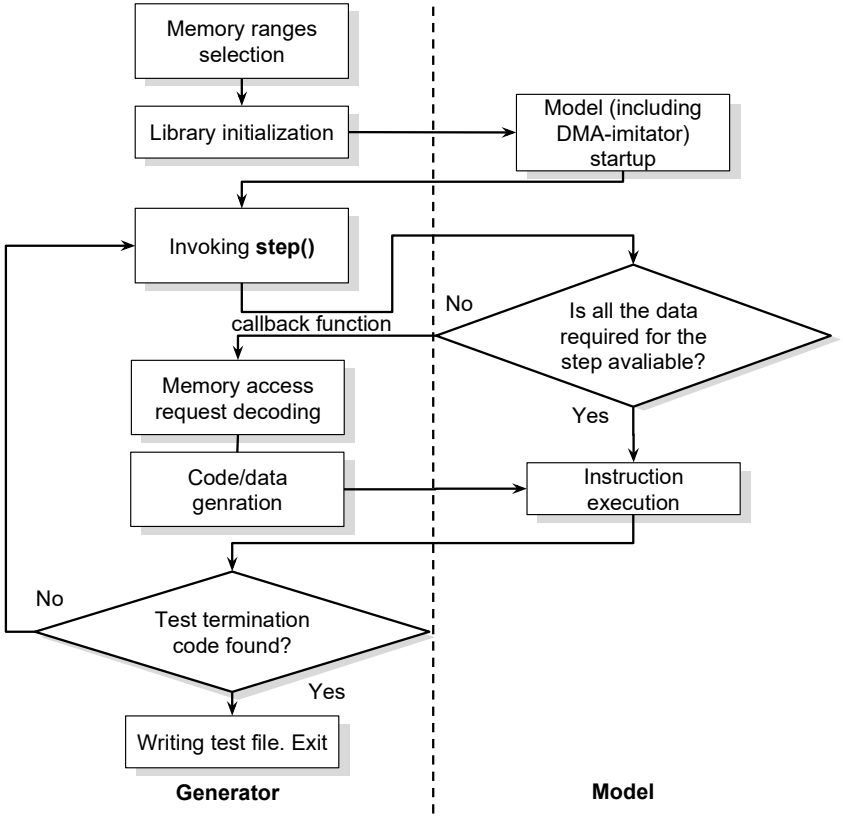


Figure 5. The control flow of the generator that employs DMA subsystem functional model.

The initialization of the DMA buffers is performed by the CPU cores causing the data for the transfer to be located at different levels of the coherent memory hierarchy that includes both caches and memory [6]. During the configuration of the imitator the specification of the operation mode and the base address of the memory to be processed are determined. The DMA exchange is performed while the CPU cores access memory regions that intersect with the DMA buffers. After the completion of the exchange the reference values are generated based on the contents of the memory final state. These values are used to perform self-checking during test execution on the target model or device. Any test produced by the generator can be executed on either the RTL model, the simulator or the FPGA-based prototype without any additional test modification. The test generator provides an opportunity to use any device connected to real southbridge instead of the DMA imitator such an Ethernet controller as a source of DMA-packets.

5. Conclusion

In this study the problem of the direct memory subsystem verification when applied to ``Elbrus'' series microprocessors has been investigated. Employment of the test generator built using the approach described in this paper allowed to find 45 bugs in three different “Elbrus” series microprocessors: 24 in a single-cores low-power CPU and no cache coherence support, 16 in a eight-core CPU supporting up to 32 core per ccNUMA system with coherent DMA and 5 in the next generation eight-core CPU with ccNUMA and updated coherence protocol. These bugs were found in spite of rigorous stand-alone verification of the DMA subsystem modules performed during the generator development. In order to enable the execution of sufficient number of tests and speeding up the development of the test generators and bug analysis a method of verification based on the replacement of DMA-capable real devices with imitator device with a simple programming interface and ability to completely consume the bandwidth of the direct memory access data path was introduced. The application of the developed method enables to achieve the operation modes of the DMA subsystem analogous to the real-world ones. The unification of the DMA imitator interface for the RTL-model, the computer complex simulator and the FPGA-based prototype allows to increase the pace of DMA subsystem tests generator development.

References

[1]. Grosso, M. et al. Functional Verification of DMA Controllers - Journal of Electronic Testing: Theory and Applications Volume 27 Issue 4, August 2011, Pages 505-516.

[2]. A.K. Kim, M.S.Mikhailov, V.M.Fel’dman. Podsistema vvoda-vyvoda dlya sistem na kristalle “MCST-4R” i “Elbrus-S” na osnove mikroskhemy kontrollera periferiinykh interfeisov. Voprosy radioelektroniki, seriya EVT, vypusk 3, 2012. (In Russian)

[3]. Gurin K.L., Meshkov A.N., Sergin A.V., Yakusheva M.A. Razvitie modeli podsistemy pamyati vychislitel’nykh kompleksov serii El’brus. Voprosy radioelektroniki, seriya EVT, 2010, vypusk 3. (In Russian)

[4]. Nohl, A., Braun, G., Schkiebusch, O., Leupers, R., Meyr, H., A Universal Technique for Fast and Flexible Instruction-Set Architecture Simulation, DAC2002, June 10-14, New Orleans, Louisiana, USA, 2002.

[5]. Frolov P.V. Generatsiya sluchainykh testov sistemnogo urovnya dlya mikroprotsessorov s arkhitekturoi El’brus. Voprosy radioelektroniki, seriya EVT, 2014, vypusk 3. (In Russian)

[6]. Isaev M.V., Polyakov N.Yu. Primenenie kesha i spravochnika DMA- obmenov v NUMA-sistemakh dlya povysheniya proizvoditel’nosti pod- sistemy vvoda-vyvoda. Pervaya vserossiiskaya nauchno-tekhnicheskaya konferentsiya Raspletinskie chteniya : sb. tez. dokl. Moskva, 2013. S. 169-170. (In Russian)

Подход к верификации модуля прямого доступа к памяти

В.Н. Куцевол <kutsevol_v@mcst.ru>,
А.Н. Мешков <alex@mcst.ru>,
М.П Рыжов <ryzhov@mcst.ru>,
П.В. Фролов <opium@mcst.ru>,
ЗАО «МЦСТ», 119334, Россия, г. Москва, ул. Вавилова, д. 24.

Аннотация. В статье описан метод верификации подсистемы прямого доступа к памяти, примененный к семейству микропроцессоров «Эльбрус». Для обеспечения возможности моделирования широкого спектра режимов работы подсистемы прямого доступа к памяти без необходимости моделировать инициализацию и работу южного моста и его контроллеров был разработан имитатор периферийных контроллеров. Имитатор был реализован как синтезируемый модуль Verilog и использовался при верификации как RTL-модели, так и основанного на ПЛИС прототипа. Интерфейс имитатора позволил встроить его вместо канала ввода-вывода, соединяющего интегрированный северный мост с процессором, что также сокращает необходимость моделировать дополнительное оборудование. Такая схема подключения имитатора также позволило использовать одну реализацию имитатора со всей серией процессоров, совместимых относительно протокола канала ввода-вывода. Модель имитатора периферийных контроллеров была также включена в функциональный симулятор вычислительного комплекса. На основе функциональной модели был разработан генератор псевдослучайных тестов, предназначенных для верификации подсистемы прямого доступа к памяти. Генератор разработан с использованием библиотечной версии функциональной модели, используемой в качестве эталонной модели во время генерации теста. Унификация программного интерфейса всех реализаций имитатора позволила исполнять тесты в неизменном в виде на функциональной модели вычислительного комплекса, RTL-модели, основанном на ПЛИС прототипе, а также произведенной микросхеме при помощи интеграции имитатора в реализованную на ПЛИС версию контроллера линка ввода-вывода. Использование описанного подхода позволило обнаружить существенное количество ошибок в разрабатываемых микропроцессорах семейства «Эльбрус».

Keywords: system verification, functional model, direct memory access, pseudorandom test generation.

DOI: 10.15514/ISPRAS-2015-27(3)-10

Для цитирования: Куцевол В.Н., Мешков А.Н., Рыжов М.П, Фролов П.В. Подход к верификации модуля прямого доступа к памяти. Труды ИСП РАН, том 27, вып. 3, 2015 г., стр. 139-148 (на английском языке). DOI: 10.15514/ISPRAS-2015-27(3)-10.

Список литературы

[1]. Grosso, M. et al. Functional Verification of DMA Controllers - Journal of Electronic Testing: Theory and Applications Volume 27 Issue 4, August 2011, Pages 505-516.

[2]. А.К.Ким, М.С.Михайлов, В.М.Фельдман. Подсистема ввода-вывода для систем на кристалле "МЦСТ-4R" и "Эльбрус-S" на основе микросхемы контроллера периферийных интерфейсов. – Вопросы радиоэлектроники, серия ЭВТ, выпуск 3, 2012.

[3]. Гурин К.Л., Мешков А.Н., Сергин А.В., Якушева М.А. Развитие модели подсистемы памяти вычислительных комплексов серии «Эльбрус». – Вопросы радиоэлектроники, серия ЭВТ, 2010, выпуск 3.

[4]. Nohl, A., Braun, G., Schkiebusch, O., Leupers, R., Meyr, H., A Universal Technique for Fast and Flexible Instruction-Set Architecture Simulation, DAC2002, June 10-14, New Orleans, Louisiana, USA, 2002.

[5]. Фролов П.В. Генерация случайных тестов системного уровня для микропроцессоров с архитектурой «Эльбрус». – Вопросы радиоэлектроники, серия ЭВТ, 2014, выпуск 3.

[6]. Исаев М.В., Поляков Н.Ю. Применение кэша и справочника DMA-обменов в NUMA-системах для повышения производительности подсистемы ввода-вывода. Первая всероссийская научно-техническая конференция “Расплетинские чтения”: сб. тез. докл., стр. 169-170, 2013.