应用于不同类型FPGA的多模式调试系统

李文昌[1]，万 理[2]，阮爱武[2]，宋子健[2]，于敦山[1]

(1. 北京大学信息科学技术学院 北京 海淀区 100871; 2. 电子科技大学电子薄膜与集成器件国家重点实验室 成都 610054)

【摘要】随着超大规模集成电路(VLSI)以及片上系统(SoC)设计的日益复杂，基于现场可编程门阵列(FPGA)的硬件仿效成为了必要环节。为解决逻辑设计下载到基于FPGA的硬件仿效器后内部节点不可视的问题，提出一种调试系统，该调试系统使用了RTL级植入调试逻辑的调试方法，统一的用户界面和软件侧底层接口，并提供了ELA模式、Scan模式和Snapshot模式。所有模式均使用统一的外部接口，使得调试系统同时适用于Altera和Xilinx的FPGA。实验结果表明，与SignalTap和ChipScope模式相比，ELA模式消耗几乎相同的资源，而Scan模式和Snapshot模式会消耗更少的FPGA资源。

关 键 词 调试; 现场可编程门阵列; 可视性; Scan模式; Snapshot模式

中图分类号 TP306 文献标志码 A doi:10.3969/j.issn.1001-0548.2014.05.009

Multi-Mode Debugging System for VLSI Designs Using Different Types of FPGAs

LI Wen-chang[1], WAN Li[2], RUAN Ai-wu[2], SONG Zi-jian[2], and YU Dun-shan[1]

(1. School of Electronics Engineering and Computer Science, Peking University Haidian Beijing 100871;

2. State key Laboratory of Electronic Thin Films and Integrated Devices, University of Electronic Science and Technology of China Chengdu 610054)

Abstract Increasingly complicated very large scale integration (VLSI) design and system-on-chip (SoC) makes Field Programmable Gate Arrays (FPGA)-based emulation necessary. As a design is downloaded into a FPGA-based emulator, invisible internal nodes of the design pose a challenge for design debugging. To address the issue, a RTL-level runtime debugging system is proposed. The user can not only select sample signals, triggering signals and statements in RTL codes, but can change triggering mode or sample window runtime as well. The proposed debugging system supports three debugging modes including embedded logic analyzer (ELA) mode, scan mode and snapshot mode. All debugging modes use a unified external interface in order to make the debugging system suitable for both Altera and Xilinx FPGAs. Experiment results show that the ELA mode consumes nearly the same resources while the scan mode and snapshot mode consumes fewer resources, compared with SignalTap and ChipScope.

Key words debugging; field programmable gate arrays; observability; Scan mode; Snapshot mode

随着VLSI和SoC设计日益复杂，目前验证和调试占据整个IC设计时间的50%～70%，成为整个IC设计过程中的瓶颈。验证和调试技术被分为3类：逻辑仿真、硬件加速和硬件仿效。与其他验证方法相比，基于现场可编程门阵列(FPGA)的硬件仿效提供了更高的性能和速度，但是当逻辑设计被下载到FPGA后，内部节点不再可见，而且FPGA只能提供极其有限的管脚以供监测内部节点，这给调试带来了很大障碍。因此，VLSI设计的日益复杂和设计周期日益缩短使得调试工具的性能要求达到了前所未有的高度。

最初的方法是使用外部测试仪，如逻辑分析仪。外部的逻辑分析仪直接连接到FPGA的引脚，以此来提供观测点。由于只有芯片的引脚可以被观测，这种方法提供的观测性很有限，并且很难定位错误。随着封装尺寸的缩小，这样的方式在连接引脚时也会产生很大的困难，因此迫切需要一种高效的调试方法来缩短调试时间，同时增加调试可视性。

为了能实现对内部信号的观测和控制，研发了许多观测内部信号的方法，如Xilinx[1-2]和Lattice[3]的FPGA提供了可配置的回读通道。然而，这样的回读方式必须在运行时暂停FPGA时钟然后做采样，这就要求被测设计有暂停和单步调试的能力。与回读相对的另一方法是直接植入调试硬件(instrument

收稿日期：2013－05－03；修回日期：2014－05－21

作者简介：李文昌(1974－)，男，博士，主要从事可编程器件的设计与测试、电源管理芯片的设计等方面的研究.

FPGA)，如植入扫描链[4-5]、嵌入逻辑分析仪(ELA)[6-8]等。植入扫描链通常是在门级进行操作，这样不易于使用，并且从FPGA接受数据会花费很长的时间。ELA方法的优点是能在不暂停设计时钟的情况下，被验证设计中暂时植入调试硬件，具有与回读相同的可观测性。但是这种植入调试硬件方式有一个较大的缺点，即每当修改了选取的信号或是重新设置了触发条件，通常要重新编译才能再次观测信号。逻辑分析仪模式的调试方式虽然能实时地观察到FPGA内部信号，但是由于FPGA片内存储器大小的限制，逻辑分析仪模式所能观测的信号个数以及采样深度都受到了限制。

近年，快照模式(Snapshot)[9]得到了改进，该调试方法是用以记录FPGA中的某些行为，并在软件仿真器中得以“重播”(replay)。Snapshot模式既拥有FPGA执行的高效性也拥有软件调试的灵活性。FPGA的内部节点中只有小部分被采样，其他的节点通过算法在软件中重构。但是，当设计对象较大时，在软件中重构的过程要消耗大量时间。

现有的商用工具也提供了强大的FPGA调试方法，如Xilinx的ChipScope模式和Altera的SignalTap模式。作为ELA模式调试方法，ChipScope模式和SignalTap模式均用FPGA片内RAM作为采样数据的缓存，因此采样点数和采样深度都受到FPGA资源的限制，而且它们都不支持RTL级的信号选取，用户使用也不够便捷。Synopsys模式提供了名为Identify RTL Debugger的调试工具。但是像大多数ELA一样，都不能做到完全的可视化，并且设计者常常不知道如何设置触发条件来发现设计的错误。

本文提出了一种新型的调试系统。该调试系统支持3种调试模式，包括嵌入式逻辑分析仪(ELA)模式、扫描链(Scan)模式和快照(Snapshot)模式。文中将这3种模式与Xilinx的ChipScope和Altera的SignalTap进行了对比分析。ELA模式可以实时监测FPGA内部信号，但需要消耗更多的资源；Scan模式可以节省大量资源，但是需要暂停设计时钟；Snapshot模式可以节省资源而且不需要暂停设计时钟，但是软件重构会需要大量的时间。因此，用户可以根据不同的需求选择相应的模式，比文献[10]具有更多的模式选择。

本文调试系统还提供了触发类型和条件的动态配置，换句话说，可以改变触发类型和条件，而且不需要重新编译逻辑设计。此外，该调试系统的通信通道被设计成Virtual JTAG形式，这使得该系统可以适用于不同类型的FPGA，即Xilinx的FPGA或Altera的FPGA。在ELA模式和Snapshot模式下，调试系统能够在不暂停内部时钟的情况下观察内部的信号，因此，本文调试系统实现了全加速全可视。

1 调试系统

本文调试系统包括软件部分、两种可供选择的通信通道和硬件部分，结构如图1所示。软件部分在PC机上运行，可以对系统进行控制，以及对逻辑设计植入调试硬件。硬件部分负责DUT接口和各种类型调试模式的实现。通信信道链接在软件和硬件之间，负责数据信号和控制信号传输。

本文调试系统的软件部分可以在一个统一的用户界面提供3个调试模式和底层接口。一旦设置了调试模式，在底层硬件部分会根据相应的调试模式产生相应的硬件结构和底层硬件开发包。

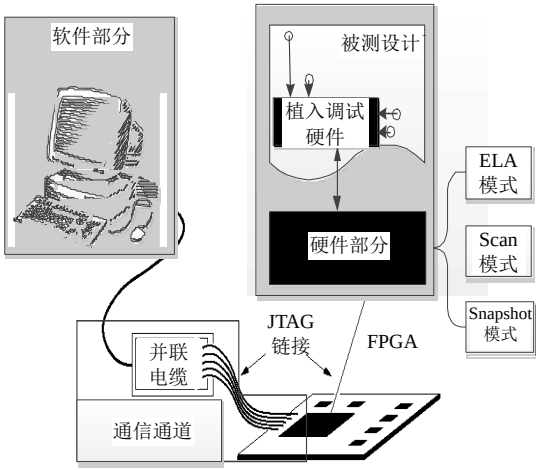

图1 调试系统的结构

1.1 软件部分

为方便用户操作，本文设计集成了软件部分所有功能的软件集成开发环境(IDE)，如图2所示。主界面是调试系统软件的调用核心，除了具有基本的代码编辑器、工程管理器、信息提示系统外，还拥有调试系统独有的功能，即可以进行Verilog代码编写与编译、建立调试工程，还可以在RTL级代码设置调试触发信号、显示波形等[11]。

整个软件系统分为4大部分：IDE控制器、RTL用户界面(UI)、后台逻辑植入部分和运行控制与显示部分。IDE控制器具有基本IDE的操作控制，包括建立工程、读入Verilog文件、查找信号、撤销操作等；RTL UI交互界面可使用户在RTL级源代码中选择观测点、设置触发信号、选择采样时钟等工作；后台逻辑植入部分负责根据RTL UI用户选择结果进

行相应的代码植入；运行控制与显示部分则负责实时控制与波形显示。

通过JTAG通道，发送数据到硬件部分的采样控制模块，控制触发条件和触发类型。运行控制与显示部分的数据流分为控制数据流和采样数据流，TCP/IP协议实现运行控制与显示部分和硬件之间的数据通信。

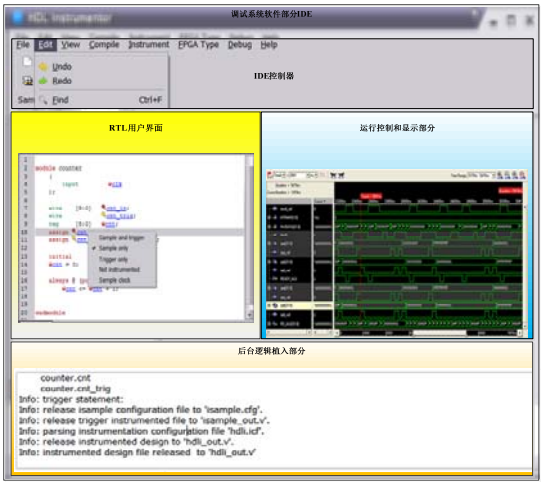

图2　软件部分集成开发环境(IDE)

1.2　硬件部分

调试系统硬件部分的总体结构如图3所示，包括：被测设计(DUT)、时钟管理模块、采样模块和触发编程模块。

事实上，被测设计是用户的原始逻辑设计，独立于调试系统。该模块中，软件部分将写入和控制植入逻辑。触发节点、观测节点和采样时钟节点分别连接到编程触发模块、采样模块和时钟管理模块。

时钟管理模块是调试系统硬件部分的时钟控制单元，负责管理和分配编程触发模块的采样时钟。

触发编程模块履行3个职责：触发条件的动态配置、设置触发的节点以及向采样模块提供采样命令信号和采样触发信号。触发节点的宽度是用户选择触发信号的总宽度。采样触发信号作为采样模块的采样信号触发条件，是触发编程模块的整体触发信号。采样模块根据采样触发信号对逻辑设计的相应节点进行采样操作。采样命令信号能确保采样模块根据相应的调试模式进行采样。将触发虚拟JTAG接口模块(VJI)嵌入到触发编程模块，作为与VJI通道层进行通信的接口。

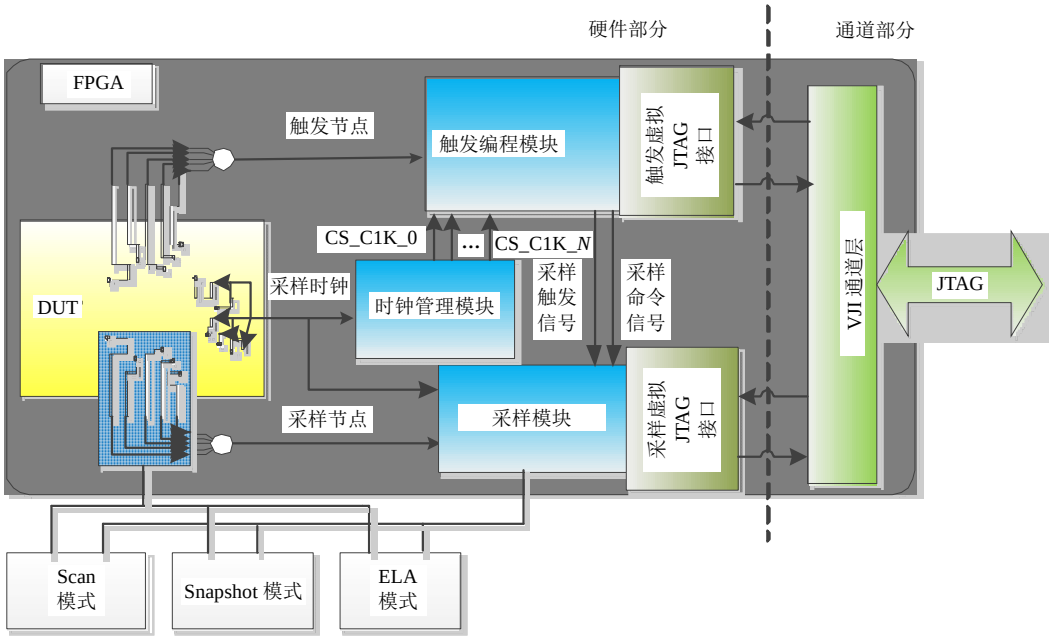

图3　硬件部分的结构

采样模块负责监控逻辑设计内部节点的信号。采样模块将根据采样命令调整信号的采样模式，并根据采样触发信号决定是否对相应模式的信号进行采样。将采样虚拟JTAG接口模块嵌入到采样模块中，在采样完成后，通过采样虚拟JTAG接口[12]和VJI通道层，将采样数据发送到软件部分。

每个调试模式实际上是用不同的方式对内部节点进行采样。调试系统统一封装采样模块，不同模式的采样模块全做成统一的外部接口，只通过更换采样模块，不修改其他调试组件，便可实现不同的模式，可以确保支持各种调试模式。因为不同模式通道层的功能将不改变，所以只需要调整软件部分的应用层即可。

1.3　通信通道

调试系统的通信层是调试系统的软件部分和硬件部分之间的通信接口，其结构如图4所示。

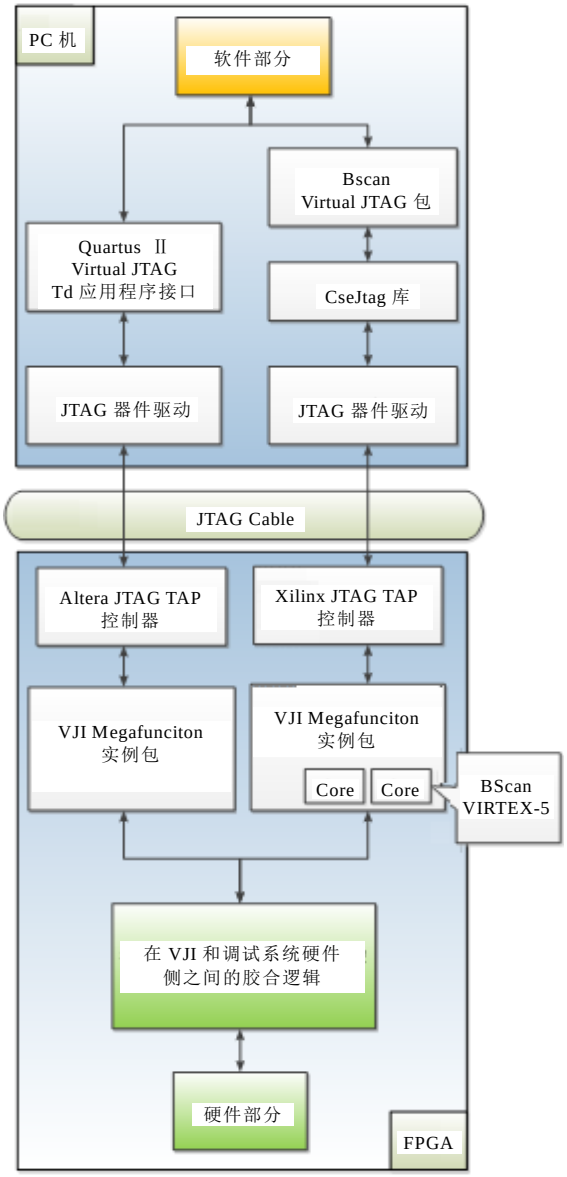

图4 通信通道的结构

Altera公司提供了虚拟JTAG接口，而Xilinx公司只提供了BScan(boundary Scan)工具[13]。如果使用虚拟JTAG接口作为调试系统的基本结构，将BScan工具软件部分和硬件部分都封装成VJI形式，那么调试系统的系统软件之间的界面设计和系统硬件方面将只需要按照VJI模型，软件部分和硬件部分不必因为不同类型的FPGA而重新设计。

由于Altera FPGA内置VJI接口，本文将根据Altera VJI的接口形式设计并实现Xilinx FPGA的软硬件部分。该软件侧的主要任务是把CseJtag库封装成BScan VJI集合，即将CseJtag库的4组命令集(csejtag_session、csejtag_target、csejtag_db和csejtag_tap)一步一步地封装成合适的函数来操纵虚拟JTAG。硬件侧的主要任务是将BScan_VIRTEX-5原语封装成类似VJI的形式，使其具有相同的硬件接口。因此，VJI的形式使得调试系统能适用于Altera和Xilinx的FPGA。

2 调试系统工作流程

调试系统的工作流程如图5所示。首先，IDE控制部分创建一个新的工程，读取Verilog源文件，编译Verilog源代码，标注编译后的RTL级代码，产生RTL用户界面。然后，用户选择采样时钟信号，选择的采样信号设定触发信号以及直接在RTL的进行其他操作。当然，本文还提供了传统的信号方式。在用户选择触发信号和采样信号后，RTL用户界面会产生hdli.icf文件，后台植入部分将根据hdli.icf文件植入硬件调试代码到Verilog源代码，而由此产生了新的文件，该文件包含了Verilog文件与调试代码。在综合之后，通过JTAG通道，将修改后的逻辑设计下载到FPGA开发板中。运行控制与显示部分将发送控制信号，通过通道层将接收到的数据信号处理并转换为波形，实时地显示在波形窗口里。

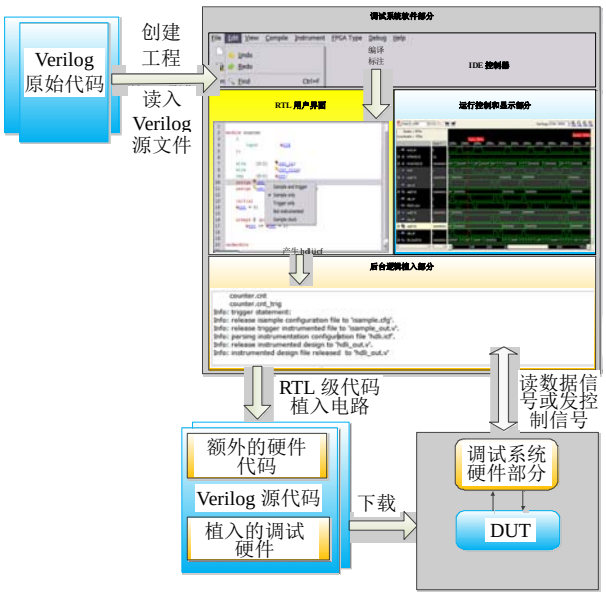

图5 调试系统的工作流程

3 实验结果

为评估系统调试，本文使用来自opencore的FPU作为测试实例。使用ELA模式、扫描模式和快照模式向Verilog源代码植入调试硬件，用Altera的Quartus II 9.1综合后，在Altera的EP2S60F1020C4上映射，与SignalTap模式进行资源比较，实验结果如表1所示。同样，使用ELA模式、扫描模式和快照模式向Verilog源代码植入调试硬件，用Xilinx的ISE12.3综合后，在Xilinx公司的XC5VLX110T上映射，与ChipScope模式进行资源比较，实验结果如表2所示。

表1　SignalTap、ELA、Scan模式和Snapshot模式资源比较

Original					SignalTap			ELA			Scan			Snapshot		
组合逻辑资源/个	寄存器/个	存储容量/KB	采样宽度/bit	采样深度/KB	组合逻辑资源/个	寄存器/个	存储容量/KB	组合逻辑资源/个	寄存器/个	存储容量/KB	组合逻辑资源/个	寄存器/个	存储容量/KB	组合逻辑资源/个	寄存器/个	存储容量/KB
3 332	628	0	16	1	3 585	1 161	16	3 602	872	16	4 754	643	0	3 452	1 184	0
				2	3 586	1 163	32	3 607	920	32						
				4	3 589	1 169	64	3 620	1 010	64						
			32	1	3 602	1 353	32	3 620	921	32						
				2	3 604	1 354	64	3 626	1 019	64						
				4	3 612	1 360	128	3 632	1 230	128						
			64	1	3 646	1 739	64	3 713	1 020	64						
				2	3 648	1 745	128	3 720	1 231	128						
				4	3 650	1 747	256	3 731	1 442	216						
			128	1	3 722	2 508	128	3 937	1 280	128						
				2	3 721	2 513	256	3 945	1 594	216						
				4	3 727	2 517	512	3 956	2 402	512						

表2　ChipScope、ELA、Scan模式和Snapshot模式资源比较

Original					ChipScope			ELA			Scan			Snapshot		
查找表数量/个	寄存器/个	块存储器/KB	采样宽度/bit	采样深度/KB	查找表数量/个	寄存器/个	块存储器/KB	查找表数量/个	寄存器/个	块存储器/KB	查找表数量/个	寄存器/个	块存储器/KB	查找表数量/个	寄存器/个	块存储器/KB
8 625	512	0	16	1	8 844	788	18	9 323	695	18	9 834	533	0	8 781	1 276	0
				2	8 841	790	36	9 929	700	36						
				4	8 848	796	72	10 920	714	72						
			32	1	8 873	855	36	9 750	700	36						
				2	8 872	856	72	10 656	706	72						
				4	8 878	862	144	12 532	712	144						
			64	1	8 947	1 005	72	10 811	801	72						
				2	8 953	1 005	144	12 528	808	144						
				4	8 955	1 011	288	16 103	819	288						
			128	1	9 124	1 347	144	13 051	1 060	144						
				2	9 137	1 346	288	16 280	1 069	288						
				4	9 140	1 352	576	23 307	1 079	576						

3.1　调试系统的ELA模式

从表1可以看出，调试系统ELA模式比SignalTap模式消耗更多的组合逻辑资源，但相应消耗更少的寄存器资源，这两种模式所消耗的存储器资源完全相同。总体看，本调试系统的ELA模式与SignalTap模式所消耗的资源几乎相同。

从表2中可以看出，调试系统ELA模式与ChipScope模式的资源消耗种类有所不同，ChipScope模式占用块存储器(BRAM)的大小与所选采样宽度与采样深度成正比；而ELA模式则不消耗BRAM资源，但却消耗更多的寄存器资源，寄存器消耗的增加量同样与所选采样宽度与采样深度成正比。1 KB的BRAM可以等效成1 024个寄存器，如果把ChipScope模式的BRAM资源折算成寄存器资源消耗，可以得出ELA模式消耗的寄存器资源更少，同时ELA模式消耗的LUT资源也更少。总体看，本调试系统的ELA模式资源消耗较ChipScope模式略少。

3.2　调试系统的Scan模式

对Scan模式进行理论分析，并中和文献[9]可以得出，Scan模式资源的增加主要是增加了扫描寄存器前的MUX资源，如一个有100个寄存器的设计，按照理论分析的结果，对该设计进行Scan模式植入后会增加1～4倍资源消耗，及增加100～400个查找表(LUT)的资源消耗。

从表1可以看出，Scan模式比原始设计增加1 422(4 754–3 332)个逻辑单元，即Scan模式资源消耗比原始设计的资源消耗约增大2.26倍(1 422/628)。从表2同样可以看出，Scan模式比原始设计增加1 209(9 834–8 625)个LUT，即扫描链模式资源消耗比原始设计的资源消耗约增大2.4倍(1 209/512)原寄存器资源消耗。由于Scan模式有部分控制电路，所以寄存器消耗比原设计略微增加。

3.3 调试系统的Snapshot模式

从Snapshot模式的理论[14]分析中可以得出，Snapshot模式资源的增加主要是对每个原设计的寄存器需要额外再添加寄存器，并把所有添加的寄存器构成一条寄存器链条，因此Snapshot模式主要增加了寄存器资源。

从表1可以看出，Snapshot模式比原始设计增加556(1 184–628)个寄存器单元，即Snapshot模式资源消耗比原始设计的资源消耗约增大0.89倍(556/628)。从表2同样可以看出，Snapshot模式比原始设计增加764(1 276–512)个寄存器单元，即Snapshot模式资源消耗比原始设计的资源消耗约增大1.4倍(764/512)。

由于Snapshot模式有部分控制电路，所以组合逻辑的消耗比原设计略微增加。因为该模式能捕获逻辑设计的所有寄存器，快照模式是不受采样节点限制的。快照模式的缺点是需要软件重构的信号，这个时间取决于编译器的性能和采样次数的多少。因此，用户可以根据需求选择不同的调试模式。

4 结 论

本文调试系统能够提供不同的调试模式去满足用户不同的需求。此外，通信层使用虚拟JTAG接口，实现了接口的统一，因此，该调试系统使得VLSI超大规模集成电路的调试应用于不同的平台，如Altera FPGA和Xilinx FPGA。该调试系统根据设置的触发条件不仅有信号触发模式，而且有语句触发模式。

同时，由于软硬件部分和通道部分的支持，调试系统具有触发条件动态配置的功能，即用户可在不重新编译设计的前提下完成对触发条件的配置和触发种类的选择。实验结果表明，与SignalTap模式和ChipScope模式消耗的资源相比，ELA模式消耗的资源相当，而Scan模式和Snapshot模式则会消耗更少的资源。

参 考 文 献

[1] Xilinx Inc. Xilinx application note XAPP138 virtex FPGA series configuration and readback[EB/OL]. (2001-11-01). http://www.xilinx.com/support/documentation/application_notes/xapp138.pdf.

[2] Xilinx Inc. Xilinx application note XAPP015 using the XC4000 readback capability[EB/OL]. (1994-10-01). http://www.xilinx.com/support/documentation/application_notes/xapp015.pdf.

[3] Lattice Technologies. Lattice application note ORCA series 4 field-programmable gate arrays[EB/OL]. (2000-04-10). http://www.latticesemi.com/dynamic/view_document.cfm?document_id=3978.

[4] WHEELER T, GRAHAM P, NELSON B, et al. Using design-level scan to improve FPGA design observability and controllability for functional verification[C]//Proceedings of 11th International Conference on Field-Programmable Logic and Applications. Berlin: Springer-Verlag, 2001: 483-492.

[5] HUANG Y, TSAI C C, MUKHERJEE N, et al. On RTL scan design[C]//International Test Conference. Piscataway, NJ: IEEE Press, 2001: 728-737.

[6] KNITTEL G, MAYER S, ROTHLAENDER C. Integrating logic analyzer functionality into VHDL designs[C]// International Conference on Reconfigurable Computing and FPGAs. [S.l.]: IEEE Press, 2008: 127-132.

[7] MAVROIDIS I, PAPAEFSTATHIOU I. Accelerating emulation and providing full chip observability and controllability[J]. IEEE Design & Test of Computers, 2009, 26(6): 84-94.

[8] GRAHAM P, NELSON B, HUTCHINGS B. Instrumenting bitstreams for debugging FPGA circuits[C]//The IEEE Symposium on Field-Programmable Custom Computing. Piscataway, NJ: IEEE Press, 2001: 41-50.

[9] CHUANG Chin-lung, LU Dong-jung, LIU Chien-nan. A snapshot method to provide Full visibility for functional debugging using FPGA[C]//The 13th Asian Test Symposium. Washington DC: IEEE Computer Society, 2004: 164-169.

[10] LI Wen-chang, SONG Zi-jian, RUAN Ai-wu, et al. A two-mode debugging system for VLSI designs using Xilinx FPGA[C]//The 3rd International Conference on Computational Problem-Solving(ICCP). Washington DC: IEEE Computer Society, 2012: 84-88.

[11] RUAN Ai-wu, LI Ceng-qi, SONG Zi-jian, et al. The third generation verification technology based SOC debugging [C]//The 2nd International Conference on Computational Problem-Solving(ICCP). Washington DC: IEEE Computer Society, 2011: 109-114.

[12] Altera Inc. Virtual JTAG megafunction user guide.pdf [EB/OL]. (2008-04-06). http://www.altera.com/literature/ug/ug_virtualjtag.pdf.

[13] Xilinx Inc. Virtex-5 FPGA configuration user guide.pdf [EB/OL]. (2010-09-06). http://www.xilinx.com/support/documentation/user_guides/ug191.pdf.

[14] Cadence Inc. Incisive palladium datasheet[EB/OL]. (2003-10-01). http://www.cadence.com/datasheets/Incisive Palladium_ds.pdf.

编 辑 张 俊