doi:10.3772/j.issn.1002-0470.2024.08.004

X-Debugger:基于 FPGA 的扫描调试器设计及实现①

李小波② 唐志敏

(处理器芯片全国重点实验室(中国科学院计算技术研究所) 北京 100190)
(中国科学院大学计算机科学与技术学院 北京 100049)

摘 要 针对芯片硅后调试面临内部信号可观测性差、可控制性弱、内部状态不易恢复重建等问题,本文设计和实现了一款基于现场可编程门阵列(FPGA)的快速扫描调试器 X-Debugger。该调试器复用传统可测试设计(DFT)扫描链路逻辑,在芯片的设计阶段插入基于功能模块前导码的扫描控制电路,实现了芯片内部各数字逻辑模块信号 100% 可见;通过基于 FPGA 的扫描调试器 X-Debugger 可以快速完成芯片内部寄存器状态获取和修改,并结合硬件加速器可以完成芯片内部逻辑状态的快速重建,从而形成硅后调试闭环。在某处理器芯片硅后调试实践中的结果表明,对于小于 100 万触发器的功能模块可以在 1 s 内完成内部状态获取、修改和重建,全芯片通过 X-Debugger 内部信号获取和重建小于 1 min,极大提高了该处理器芯片的硅后调试效率。

关键词 硅后调试;现场可编程门阵列(FPGA);扫描链;寄存器回读;状态重建

随着集成电路工艺的进步,芯片集成度不断提高,芯片设计也愈发复杂。复杂芯片在硅后由于可观测性差和可控制性弱,内部状态不易获取和重现,硅后调试也变得十分困难[1]。如何提高芯片的硅后调试能力,是一个亟待解决的问题。

基于调试引脚(debug IO)的信号选择技术把芯片内部部分关键信号和状态引出到外部 IO 引脚上,然后可以外接示波器和逻辑分析仪等设备查看芯片内部部分关键信号的状态。但由于调试引脚数量有限和内部状态信号部分选择原因,只能观测芯片内部部分状态,且不能对芯片内部信号进行修改和控制。

基于扫描链(scan chain)[2]的调试技术是一种成本较低、效果较高的芯片硅后测试技术,被广泛应用于芯片的可测试设计中[3]。其基本原理是用扫描触发器替代原有的功能触发器,并用扫描链将芯片内部扫描触发器连接起来,在进入扫描模式后,扫描时钟(scan clock,SCLK)和扫描输入(scan in,SI)每个时钟移入(shift in) 1 位(bit)数据,同时芯片内部也会移出(shift out) 1 bit 数据,从而达到对芯片内部触发器状态的获取和控制。

随着芯片规模增大,传统基于扫描链的调试技术由于需要串行移位,获取中间所需的某功能模块触发器的状态费时且效率较低。此外获取到的芯片内部的状态数据,都是无意义的 0/1 序列,需要根据触发器在扫描链中的相对位置倒推复现该触发器及相关逻辑的内部状态。

有很多为硅后调试专门插入的调试逻辑技术[4],如 Trace buffer[5]、IFRA(instruction footprint recording and analysis)[6]等,这些技术主要局限于只能获取芯片内部部分关键信号和状态,无法达到 100% 可见[7-8]。此外只能对信号进行观测,无法控制和修改芯片内部状态,无法辅助芯片的硅后快速调试和问题定位。

① 国家自然科学基金(61732018,61872335,61802367)资助项目。
② 男,1982 年生,博士生;研究方向:计算机系统结构,处理器设计与验证;联系人,E-mail:lixiaobo17b@ict.ac.cn。
(收稿日期:2023-05-06)

本文实现的基于现场可编程门阵列(field programmable gate array,FPGA)的扫描调试器利用芯片原有的扫描链逻辑,在引入开销不大的情况下,快速获取芯片内部信号,让芯片内部数字逻辑状态100%可见,并可以实现芯片内部触发器的100%可控制修改,从而辅助芯片硅后软硬件调试。

本文基于 FPGA 实现了一款快速的扫描调试器 X-Debugger,主要贡献如下。

(1)提出了基于功能模块前导码的扫描链插入技术,在芯片设计阶段对各功能模块插入前导码和扫描控制选择逻辑,方便按功能模块获取芯片内部触发器的状态。

(2)基于 FPGA 实现了一款高速扫描调试器,包括相应配套调试软件,可以快速获取芯片内部触发器状态,并在恢复或修改芯片原始状态后可以续运行。

(3)引入基于功能模块扫描网表或全芯片网表的硬件加速器(emulator)实现,不需要任何扫描链相对位置信息,可以较快完成芯片内部状态的重建(replay),从而完成芯片调试闭环,快速定位硅后软硬件问题。

1 X-Debugger 扫描设计

传统的扫描链插入技术是现代芯片可测试设计(design for test,DFT)中的成熟技术,本文不作详细介绍。本文提出和实现 X-Debugger 调试器系统,主要包括基于功能模块前导码的扫描链控制和选择技术,可以快速完成芯片内部各个功能模块信号获取、实现内部数字逻辑 100% 可见;所有内部触发器100%可控,可以实现芯片原始状态恢复或修改并继续后续运行;实现寄存器信号回读和网络传输;基于加速器和 X-Debugger 完成芯片内部状态的快速重建。

1.1 功能模块扫描设计

芯片设计可以划分成多个功能模块,为了满足后续扫描调试技术要求,在芯片内部的功能模块划分时,需要在设计阶段满足 2 个要求。

(1)各个模块的输入信号需要触发器锁存。这主要是为了确保功能模块的内部状态在扫描模式改变时,不会影响到与之交互的其他功能模块后续的正常工作状态。

(2)扫描链按照功能模块进行连接。如图 1 所示,在功能模块 0(Function 0)的触发器完成扫描链后,再将整个模块的扫描输出端口(scan out,SO),链接到下一个功能模块 1(Function 1)的扫描输入 SI,让 Function 1 的 SO 再连接下一个功能模块的扫描输入 SI,以此类推,从而完成全芯片扫描链基于功能模块的串行连接。

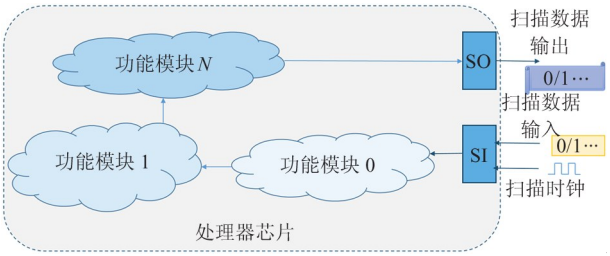

图 1 功能模块扫描链连接

1.2 功能前导码及扫描控制逻辑插入

基于功能模块前导码的扫描选择技术,可以将芯片中的各主要功能模块状态,在扫描获取控制器(scandump controller)的控制下,方便快捷地扫描到输出端口,从而可以更快获取所需模块的状态。其主要逻辑如图 2 所示。

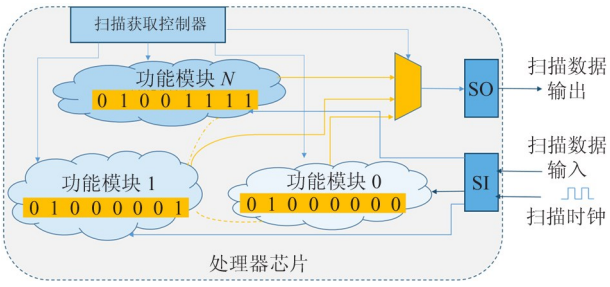

图 2 功能模块前导码选择扫描

芯片中每个功能子系统,通过在芯片扫描设计阶段添加一组功能模块前导码,每个扫描功能子系统有唯一标识的前导编码(preamble)。它是一个固定的 8 bit 序列,基于该功能模块扫描链的长度信息通过哈希(Hash)算法生成,方便校验该模块移出数据的正确性。每次从正常工作模式进入扫描获取

(Scandump)模式,移出数据都是从该序列开始的,根据该信息可以快速检查每个模块扫描输出数据是否完整。同时它还可以作为前导码区分移出的数据来自哪个功能模块,方便后续输出数据对齐和重放,从而快速完成内部功能模块状态的提取和检测。每次从 Scandump 模式恢复到正常工作状态时,又自动恢复该前导序列,该前导序列不影响处理器各个模块正常功能。

在 Scandump 功能模块划分和扫描链串联设计方面,需要遵循芯片实际应用数据通路(data-path)顺序。比如功能模块 X 数据,会后续传递给模块 Y,在设计扫描链路时,尽量将模块 X 的下一级链接到 Y 上,从而可以尽快在 Scandump 模式下获取数据通路前后级的信号状态。

基于功能模块前导码选择的扫描技术可以有效地解决芯片调试的断链故障。传统扫描技术将所有模块都串联在同一个扫描链上,由于芯片设计规模巨大、寄存器众多,全扫描链长度可能达到几千万甚至上亿个寄存器。一旦发生断链故障挂死(hang),其他模块的状态就无法扫描输出到调试端口,如果采用基于功能模块前导码的扫描选择技术,就可以旁路(bypass)故障模块,除故障模块本身外,其他功能模块不会受到影响。

另一方面,基于功能模块前导码选择技术可以快速获取调试所需模块的状态。而在调试时,可能只关注某个子系统或者某个外设接口的状态,比如怀疑芯片设计中的外围组件高速互联(peripheral component interconnect express,PCI-E)外设接口异常,只需要获取(dump)PCI-E 功能模块相关寄存器,直接选择该模块扫描信息输出到调试端口,从而节约状态获取和重建时间,提高芯片的硅后调试效率。

1.3　Dummy 寄存器融合前导码

由于芯片设计规模巨大,逻辑上可以将任何一个功能模块信号通过 Scandump 控制逻辑直接旁路到输出端口。但是物理上来讲,一些功能模块离芯片的扫描输出端口 SO 引脚的物理距离较远,时序较难收敛,或者即便收敛,也会降低扫描电路工作的最高频率。

一种常用方法是在扫描链的路径上插入空寄存器(dummy registers)来改善时序(timing),本文结合功能模块前导码,将 8 位功能前导码寄存器作为 Dummy 寄存器,这样既满足了物理设计实现时远距离时序要求,又可以完成模块区分和前导码输出及数据对齐。

1.4　信号捕捉流程

插入基于功能前导码的芯片,其内部数字逻辑状态捕捉(capture)流程如图 3 所示。

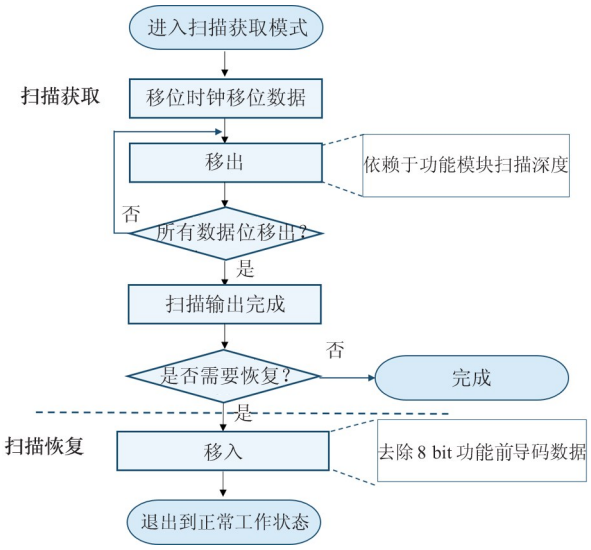

图 3　信号扫描获取和恢复

Scandump 工作流程分为以下 8 个步骤。

(1)通过特定的配置,可以让芯片进入 Scandump 模式。

(2)芯片进入 Scandump 模式后,所有的芯片逻辑功能不再运行,各个信号的当前状态进入 Scandump 链。

(3)通过 Scandump 控制器选择需要扫描移出的功能模块和移入的模块。

(4)当扫描获取时钟(Scandump Clk)有时钟时,在每一个有效的时钟边沿,会产生 2 个行为:

1)Scandump 移出从 Scandump chain 移出1 bit 数据;

2)Scandump 移入向 Scandump chain 移入1 bit 数据。

(5)通过上述流程,当 Scandump 链的全部数据移出后,就获取了芯片内部所有时序寄存器信号的

状态信息。

(6)如果需要恢复芯片原始状态,可以继续进行 Scandump shift 模式。

(7)将移出的数据序列除去 8 bit 的模块前导码,继续重复移位回该功能模块。

(8)完成移位后,退出扫描获取模式(Scandump mode),恢复到正常状态,使能(enable)正常时钟逻辑(logic)后,处理器芯片可以恢复到原始状态(进入 Scandump mode 前的原始工作状态)。

如果需要修改芯片的状态,可以将第 7 步的移入数据序列作相应调整和修正,在移入完成后,就完成芯片内部寄存器值的修改和控制,从而让芯片进入一个新的状态开始后续运行。

Scandump 导出的 Scandump 数据是一个二进0/1序列,不具备可读性。只有将它们转换成内部信号的状态表示,才能发挥作用。

1.5 基于硬件加速器的芯片内部状态重现

常规的芯片内部信号提取到的 0/1 序列,需要结合触发器在扫描链的位置信息,在芯片物理实现后,触发器的位置信息已经被打乱。比如芯片内部直接存储访问(direct memory access,DMA)功能模块的一个 64 bit 的数据寄存器,这 64 bit 可能并不是连续的依次排列在扫描链上,可能被分散在扫描链上不同位置,需要复杂的反标倒推才能把这些信息按从高位到低位的 bit 顺序拼凑提取出来。而寄存器间的组合逻辑信号状态恢复更加困难,这是基于扫描信号恢复和重建的难点。

本文引入了基于仿真加速器的重建。仿真加速器是一种灵活可配置硬件设备,主要用于硬件加速仿真,被广泛应用于加速芯片验证流程[9],其运行速度可达 1 ~ 2 MHz。可以将芯片设计逻辑通过仿真加速工具综合,完成构建后放入到仿真加速器中,可以理解成芯片在其中的一个实现。

为了克服基于扫描信息的芯片内部状态重建难题,利用仿真加速器,实现扫描链的数据移出的逆操作,把相应信息移入等价的芯片设计功能模块中。利用仿真加速器较快的运行速度和内部信号全可见特征,可以方便地完成芯片状态的快速重建。其整体原理如图 4 所示。

首先通过主机控制 X-Debugger 调试器完成对芯片硅上(silicon)全芯片或者某个功能模块的内部扫描寄存器进行扫描输出提取。通过图 4 的工作流程,获取内部寄存器序列后,再通过以太网接口发送到主机 1 中。

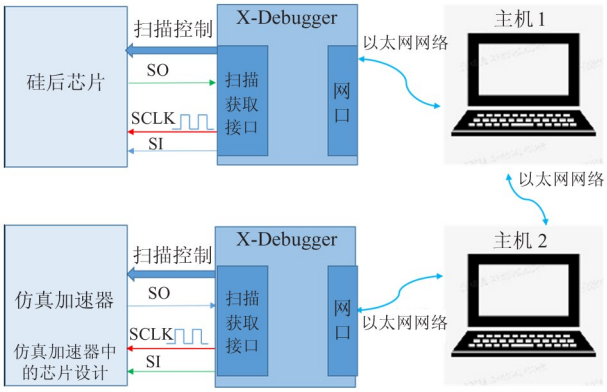

图 4 基于 Emulator 的重建

然后将获取到的内部扫描信息通过网络传送到连接仿真加速器的主机 2 上,通过同样的 X-Debugger 调试器以逆过程扫描输入将内部序列移入 Emulator 的芯片设计中。再切换回正常功能模式,通过硬件仿真加速工具,可以上传(upload)该功能模块的内部状态,并通过波形转换和查看工具可以查看芯片内部模块时序逻辑和全部组合逻辑的状态。

在芯片设计阶段,考虑到扫描链按照功能模块划分和连接,本文在内部信号重建时也可以十分灵活。如果只需要某个模块或者某几个模块,只需要把这几个模块综合实现在仿真加速器中,从而节约有限的硬件加速器资源,此外较小的规模也有利于提高硬件加速器的仿真调试速度。

2 X-Debugger 设计

为了快速并灵活控制芯片的扫描链完成芯片内部状态获取、状态恢复和状态修改,并完成基于 Emulator 的芯片状态重建,需要设计一款灵活而高效的扫描调试器。本文通过选用可编程的 FPGA,实现了 X-Debugger 调试器。从宏观层面,X-Debugger 设计需要满足 2 点需求。

(1)能够使用高速接口(如以太网、通用串行总

线)和上位机连接通讯,实现命令交互和数据交互。

(2)能够支持特殊的硬件时序(如 Scandump 时序),并可以调节其工作频率,以实现调试功能的扩展。

2.1　X-Debugger 整体架构

X-Debugger 是一款灵活而高速的扫描获取调试器,其整体架构如图 5 所示。

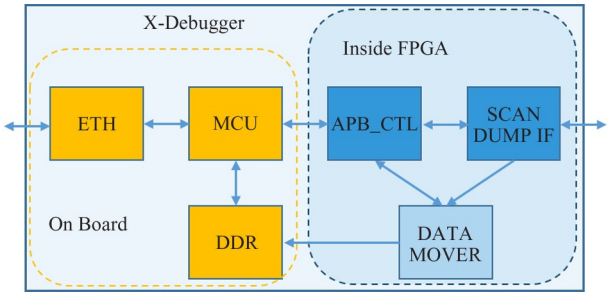

图 5　X-Debugger FPGA 系统逻辑架构

X-Debugger 调试器包括两大部分,一部分是左侧 3 个模块,主要完成与主机之间接口和数据存储转换。包含一个高速的以太网控制器(ethernet,ETH)、一个可编程微控制器(micro control unit,MCU)及高速双倍速率同步(double data rate,DDR)模块。剩下模块主要是 FPGA 的可编程部分,包括一个频率可调节的 Scandump 接口模块、一个高级外围总线(advanced peripheral bus,APB)控制模块、一个移入数据/移出数据接口模块。各模块主要功能如下。

APB_CTL:基于 APB 的控制模块,在 FPGA 内实现了 APB 总线 Slave 控制逻辑,它包含 Debugger 的控制配置寄存器,完成和 SCANDUMP IF/DATA MOVER 作控制及状态的交互。

SCANDUMP IF:产生 Scandump 的时钟 SCLK,接收芯片的 Scandump 输出数据并通过高级扩展接口(advanced extensible interface,AXI) STREAM 送到 DATA MOVER 模块,或将从主机接受到的数据移入的芯片中。

DATAMOVER:接收 AXI STREAM 信号,转换成 AXI 协议,写入到 DDR 存储中。

MCU:该 MCU 处理器包含 CORTEX-A9 双核处理器,作为设备主控,支持软件灵活可编程实现整个 X-Debugger 的控制和数据交互。

DDR:片外存储 DDR4 存储(memory),ARM 核和 FPGA 逻辑都可以访问,用于程序和数据存放。

ETH:千兆以太网控制器,和上位机通信连接,可以从网络获取数据或者发送数据给主机。

2.2　X-Debugger FPGA 实现

为了方便 X-Debugger 的 FPGA 实现,选择 Xilinx 的 ZYNQ[10] 作为 X-Debugger 的基础平台。对比传统的 MCU CHIP + FPGA CHIP 的方案,ZYNQ 有诸多优势。如内置了 ARM 硬核和常见外设(如以太网/通用串行总线)接口,这些功能模块不再需要占用可编程逻辑(programmable logic, PL)资源,且能达到较高的性能;PL 之间通过高速 AXI 连接,更容易扩展自定义外设,且性能更好。此外,还有完善的工具流程和底层库支持,加速开发迭代。

FPGA 实现主要包括 APB_CTL 控制模块和 Scandump 的接口模块。为了可以同时连接芯片和仿真加速器,本文实现了软件灵活可配置的扫描接口频率,在 FPGA 上该 Scandump IF 的扫描时钟 SCLK 最高可以工作在 200 MHz,最低通过分频可以工作在 100 kHz。APB 模块 APB 时钟的工作最高频率也可达 200 MHz。

在不考虑 DDR4、ETH、ARM 等硬核的情况下,FPGA PL 部分其面积如表 1 所示,可见整个 FPGA 的查找表(look up table,LUT)和寄存器(register)资源开销非常小。

表 1　各模块 FPGA 实现面积

模块	LUTs	REGISTERs
SCANDUMP IF	15 472	3 576
APB_CTRL	6 617	4 292
DATAMOVER	17 768	6 878
合计	39 857	14 746

2.3　X-Debugger 软件实现

在 X-Debugger 中,需要软件完成整个 X-Debugger 的控制流程, X-Debugger 整个系统包含 2 套软件:运行在主机(Host)的 Scandump 上位机工具和运行在 MCU(FPGA 内部)的嵌入式程序。它们之间的通信基于自定义的应用协议。

Scandump 通信协议是基于标准的 TCP 协议,实现了命令下发、数据捕获等通信机制。

(1)Scandump 上位机工具程序:实现 Scandump 通信协议的客户端,下发命令给 X-Debugger,从 X-Debugger 接收捕获的 Scandump 数据,并将 Scandump 数据保存成所需格式的文件。

(2)Scandump FPGA MCU 嵌入式程序:实现 Scandump 通信协议的服务器(Server)端,接收上位机下发的命令,将捕获的 Scandump 数据返回给上位机;并实现 Scandump IF 和 DATAMOVER 等模块的控制,完成 Scandump 数据捕获或者恢复(修改)数据的移入。

3 实验结果

针对某图形处理器(graphic processing unit,GPU)片上系统(system on chip,SoC),采用基于 FPGA 的 X-Debugger 调试器,可以快速完成整个 SoC 内部或者各个功能模块寄存器数据获取。该 GPU SoC 整体规模数字逻辑部分超过 10 亿门,全芯片有 4 700 多万个触发器,按照功能被划分包含 7 大功能子系统,分别为 MCU 子系统、音视频处理器子系统、DMA 子系统、Graphic 子系统、存储子系统、显示子系统和主机连接子系统,如图 6 所示。

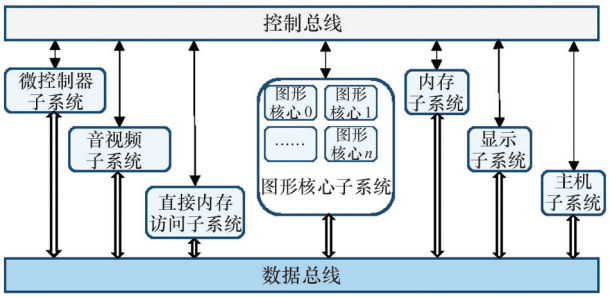

图 6 GPU SoC 功能模块划分

其中 GPU 子系统包含多个图形渲染核心(graphic core),每个核心由于设计十分复杂,规模较大,又被拆分为 4 个子功能模块,所以全芯片在扫描链设计时,划分为 38 个子功能模块,每个模块在设计阶段采用 1.2 节的基于功能模块前导码的扫描逻辑插入和选择连接。

扫描逻辑在硅上最高工作在 100 MHz,而 X-Debugger FPGA 实现可以支持 0.1 ~ 200 MHz 的扫描时钟,所以完全可以支持硅上的全速扫描信号获取和快速恢复。

表 2 实验结果表明,对于规模小于 100 万触发器的功能模块,从扫描接口上看,SCLK 工作在 100 MHz、10 ms 就可以完成触发器状态数据获取和修改(恢复),从主机下达命令,到数据回读完成整个流程时间小于 1 s,使用 X-Debugger 调试器没有任何延迟的感觉,调试数据可以实现无延迟感获取。

表 2 X-Debugger 功能子系级扫描用时

功能子系统	REGISTERs	扫描时间/ms
MCU 子系统	601 827	6
Video 子系统	1 084 292	11
DMA 子系统	170 962	2
Gcore 0(单个)	10 832 128	108
Memory 子系统	378 603	4
Display 子系统	938 724	10
Host 子系统及其他	687 590	7
合计	47 190 510	500

该芯片功能模块基于 Veloce 仿真加速器[11]的实现,工作在 1.57 MHz 左右,可以在 1 s 内完成数据移入和内部状态重建。如果采用传统根据扫描链信息倒推方式,对于 100 万触发器规模的子系统,获取期望位置某几个寄存器状态信息,需要根据触发器在扫描链上的位置信息,进行大量的组合排序,如果没有工具支持,人工操作将费时费力,效率低下且容易出错。

对于全芯片数字逻辑部分有 4 700 多万个触发器,完成全部信号的获取时间和重建时间小于 1 min。正是由于 X-Debugger 的使用,在芯片的硅后调试验证中,可以快速获取芯片内部功能模块/全芯片内部状态,完成内部状态基于仿真加速器重建,实现了芯片数字逻辑部分 100% 可见和触发器的 100% 可控制可修改,极大加快了芯片调试进程,提

升了芯片硅后调试效率。

4 结 论

芯片的硅后调试是一项重要且有挑战性的工作，本文基于传统的扫描技术，在复用芯片的可测试扫描逻辑基础上，减少了测试逻辑开销，通过芯片设计阶段插入基于功能模块前导码的扫描链路和扫描控制选择电路，开发了基于 FPGA 的快速调试器 X-Debugger，该调试器可以快速获取芯片各个功能模块内部状态。整个调试器逻辑开销小、速度快，可以通过网络接口灵活远程控制。在获取芯片状态的同时，可以完成芯片内部触发器信号的恢复和修改。并可以将回读数据通过网络实时传输到主机上，可以基于 Emulator 进行状态重建，从而快速获取处理器内部状态，实现调试闭环，极大提升了芯片硅后的调试效率。

基于 FPGA 的扫描调试器也有一些不足，如扫描串行接口内部触发器状态获取深度较大。未来将继续提升该调试器系统的性能，如通过增加多路并行扫描选择端口，可以同时灵活控制几路扫描接口，同步获取芯片内部模块状态。通过添加调试跟踪和触发逻辑，连续跟踪芯片内部连续关键状态，如程序执行状态（program counter，PC）序列流程，在 PC 执行异常时，可以自动进入 Scandump 状态，并保留现场、上报中断，这种调试技术不仅可以调试硬件电路，还可以辅助软件调试。

参考文献

[1] AGALYA R, SARAVANAN S. Recent trends on post-silicon validation and debug: an overview[C]//Proceedings of 2017 International Conference on Networks & Advances in Computational Technologies. Thiruvananthapuram, India : IEEE, 2017:56-63.

[2] KULKARNI K S, ARADHYA H. Dataset development of GPU block using scan dump for silicon debug[C]//Proceedings of the 6th International Conference on Communication and Electronics Systems. Coimbatre, India: IEEE, 2021:322-326.

[3] HUANG Y, GUO R, CHENG W T, et al. Survey of scan chain diagnosis[J]. IEEE Design and Test of Computers, 2008,25(3):240-248.

[4] RAMIREZ W, SARMIENTO M, ROA E. A flexible debugger for a RISC-V based 32-bit system-on-chip[C]// Proceedings of IEEE 11th Latin American Symposium on Circuits and Systems. San Jose, Costa Rica: IEEE, 2020:1-4.

[5] KALIMUTHU P, BASU K, SCHAFER B C. Efficient hierarchical post-silicon validation and debug[C]//Proceedings of 2021 34th International Conference on VLSI Design and 2021 20th International Conference on Embedded Systems. Guwahati, India: IEEE, 2021: 258-263.

[6] DECKER N, DREYER B, GOTTSCHLING P, et al. Online analysis of debug trace data for embedded systems [C]//Proceedings of 2018 Design, Automation and Test in Europe Conference and Exhibition. Dresden, Germany: IEEE, 2018:851-856.

[7] BASU K, MISHRA P, PATRA P, et al. Dynamic selection of trace signals for post-silicon debug[C]//Proceedings of 2013 14th International Workshop on Microprocessor Test and Verification. Austin, USA: IEEE, 2013:62-67.

[8] OMIDIAN H, HUNG E, GAITONDE D. 100% visibility at MHz speed: efficient soft scan-chain insertion on AMD/Xilinx FPGAs[C]//Proceedings of Applied Reconfigurable Computing. Architectures, Tools, and Applications: 18th International Symposium. Cham: Springer, 2022:1-16.

[9] HERBST S, RUTSCH G, ECKER W, et al. An open-source framework for FPGA emulation of analog/mixed-signal integrated circuit designs[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021,41(7):2223-2236.

[10] XILINX. ZYNQ FPGA product [EB/OL]. (2018-12-07) [2023-03-18]. https://www.xilinx.com/products/silicon- devices/soc/ zynq-7000.html#productTable.

[11] SIEMENS. Veloce strato Emulation platform[EB/OL]. (2020-11-7) [2023-3-18]. https://eda.sw.siemens.com/en-US/ic/veloce/

X-Debugger:an FPGA based scan debugger design and implementation

LI Xiaobo, TANG Zhimin

(State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

(School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049)

Abstract

Chip post silicon debugging faces challenges such as poor internal observability, weak controllability, and difficult internal state replay. This paper designs and implements a field programmable gate array (FPGA) based scan debugger X-Debugger which improves the debug efficiency dramatically. By reusing origin design for test (DFT) scan chain logic, inserting a scan control circuit based on functional module preamble during the chip design phase, the digital logic signals inside the chip are 100% visible. The scan debugger based on FPGA can quickly complete the acquisition and modification of the internal register state of the chip, and complete the replay of the internal logic state of the chip through the emulator rapidly, thus forming a debugging closed loop. The post silicon debugging practice of a processor chip shows that the function module with less than one million registers can complete the internal state acquisition and replay in 1 s, and the whole chip's internal state can be acquired and replayed in less than 1 min by X-Debugger, which greatly improves the post silicon debugging efficiency of the processor chip.

Key words:post-silicon debug, field programmable gate array (FPGA), scan chain, registers readback, state replay