WP267 (v1.0) 2007 年 8 月 15 日

Spartan–3A/3AN/3A DSP FPGA 的高级安全机制

作者： Glenn Crow

FPGA 具有轻松集成与支持新协议和新标准以及产品定制的能力，同时仍然可以实现快速的产品面市时间。在互联网和全球市场环境中，外包制造变得越来越普遍，这使得安全变得更加重要。正如业界领袖出版的文章所述，反向工程、克隆、过度构建以及篡改已经成为主要的安全问题。据专家估计，每年因为假冒产品而造成的经济损失达数十亿美元。国际反盗版联盟表示，这些假冒产品威胁经济的发展，并且给全球的消费类市场带来重大影响。本白皮书将确定设计安全所面临的主要威胁，探讨高级安全选择，并且介绍 Xilinx 的新型、低成本 Spartan™–3A、Spartan–3AN 和 Spartan–3A DSP FPGA 如何协助保护您的产品和利润。

© 2007 Xilinx, Inc. All rights reserved. All Xilinx trademarks, registered trademarks, patents, and further disclaimers are as listed at http://www.xilinx.com/legal.htm. All other trademarks and registered trademarks are the property of their respective owners. All specifications are subject to change without notice.

NOTICE OF DISCLAIMER: Xilinx is providing this design, code, or information "as is." By providing the design, code, or information as one possible implementation of this feature, application, or standard, Xilinx makes no representation that this implementation is free from any claims of infringement. You are responsible for obtaining any rights you may require for your implementation. Xilinx expressly disclaims any warranty whatsoever with respect to the adequacy of the implementation, including but not limited to any warranties or representations that this implementation is free from claims of infringement and any implied warranties of merchantability or fitness for a particular purpose.

主要安全威胁有哪些?

当今设计领域所面临的主要安全威胁包括*反向工程*、*过度构建*、*克隆*和*篡改*。

当盗窃者出于重建或重造竞争性产品并在公开市场上销售之目的而利用您的设计时，这便是*反向工程*。反向工程的作用是窃取者可以更快地完成设计，并且可以将研发成本最小化。这是电子行业自问世以来遇到的最普遍的威胁。

*过度构建*是外包业务模型的一大隐患。在这种情况下，产品经过未授权过度构建，然后在未得到原始设备制造商许可的情况下通过其他渠道销售。最大的问题是，一旦这些产品进入市场，它将造成非常严重的后果。通常，“过度构建”的产品能够以更快的速度面市，并且其销售价格也更低。

当盗窃者以相同或不同的商标复制设计、IP 或产品，这就是*克隆*。克隆者获得的明显利益是，他们无需花费任何研发成本，并且极大地加快了克隆产品的上市。

*篡改*是对原设计进行修改和 / 或替换以获取未授权服务、盗取敏感数据，或破坏应用程序。篡改是金融、防卫以及音频 / 视频媒体收费服务供应商的巨大隐患。

Spartan–3A/3AN/3A DSP FPGA 中的安全机制

用来保护 PFGA 的安全级别和类型与成本有关。首先，要认识到没有牢不可破的安全这一事实很重要。基本上，要完全阻止攻击者破解系统是不可能的。如果有人想要得到您的数据或设计，他们可采用蛮干来得到他们想要的东西。这不是一般的黑客，很可能是一个资金充裕的政府或者是一个资金充裕的竞争对手，因此，您不可能创建一个永远不会被破解的解决方案，而是创建能够充分保护您的设计免受克隆者、过度构建者、篡改者以及反向工程者给您带来的常见威胁的解决方案。当您想到安全时，您需要考虑满足您的需求的解决方案。如果您的产品值 10 美元，您只能在该价格范围内为该系统提供适当的安全性，而不是可能价值 10,000 美元的系统。这是您需要做的评估。完成评估后，您可以在该评估的基础上确定您希望实现的产品和安全级别。Xilinx 提供各种解决方案供您选择来解决您的安全问题，从简单解决方案到复杂解决方案。“利用 Spartan–3 系列 FPGA 实现安全解决方案”白皮书 (WP266) 讨论了在 Spartan–3 系列内实现的基本安全解决方案。

本白皮书探讨更高级的技术，比如：

- 比特流发生器的安全级别
- 主动防御（JTAG 边界扫描）
- 比特流验证（循环冗余校验（CRC））
- 高级数据操作

除了 Spartan 产品以外，Xilinx 还提供利用 Virtex™ FPGA 产品实现的更为先进的解决方案。

比特流发生器的安全级别

在设计的测试和调试阶段，设计者可以决定将内部配置访问端口（ICAP）或 ChipScope™ Pro 分析器核留在设计中，来方便设计投产后可能进行的维护或随机检查。一些软件工具（如 ChipScope Pro 分析器）需要这些宏命令来读取内部逻辑的状态。虽然这为设计者带来了方便，但会留下安全漏洞。

比特流发生器基于名为 NCD 文件的物理实现文件的内容创建配置 .bit 文件。.bit 文件规定已编程 FPGA 的行为。比特流发生器包括许多选项，其中一个选项是安全级别设置。比特流发生器有四个安全级别设置，第一个是默认值，其余三个选项则提供额外的安全保护。如表 1 所示，读回命令操作可以被完全禁用，或者仅限于通过 ICAP 从 FPGA 应用进行内部访问。

表 1: **比特流发生器安全级别设置**

安全级别	描述
无	默认值。无限制访问所有配置和读回功能。
1 级	禁用两个 SelectMAP 或 JTAG 端口（外部引脚）的所有读回功能。允许通过 ICAP 读回。
2 级	禁用所有端口上的所有读回操作。
3 级	禁用所有配置和 JTAG 端口上的所有配置与读回功能。3 级中唯一可以发出并执行的命令（读回和配置）是 REBOOT。它会擦除器件的配置。这与启用器件上的 PROG_B 引脚具有相同的功能，除了它是在器件内进行的。

了解所有比特流发生器选项的详细信息，请参照 “Spartan–3 系列配置用户指南 (UG332)”。采用以上安全设置，1、2 和 3 级，将会阻止以下需要 ICAP 基元的任何解决方案。

主动防御（JTAG）

一个普遍的问题是任何带有 JTAG 接口的器件都易受反向工程的攻击。通过采用边界扫描链，JTAG 也可以用于对系统、器件、IP 或标准产品进行反向工程。这需要攻击者拥有充足的资金、渊博的知识和熟练的技术，以及相应的设备与时间。组织、竞争对手或政府都在试图了解一个产品是怎样工作的并且将很有可能试图降低成本或增加功能。本节讨论将不同特性整合到您的设计中来检测和防止 JTAG 反向工程的方法。

JTAG 边界扫描起初是为协助测试和调试 PCB 上的 I/O 连接功能而设计的，后来是用于将逻辑整合到芯片内部。在使用 INTEST 命令和边界扫描功能时，您可以将数据移动到模块或 IC 中，然后用时钟记录 IC 来读回结果数据。该操作可以为技术娴熟的用户提供 IC 或模块中的架构或逻辑。如图 1 所示，这也是一种对设计或系统进行反向工程的方法。因此，未授权使用 JTAG 端口对于有些用户及其产品的安全来说是一个隐患。

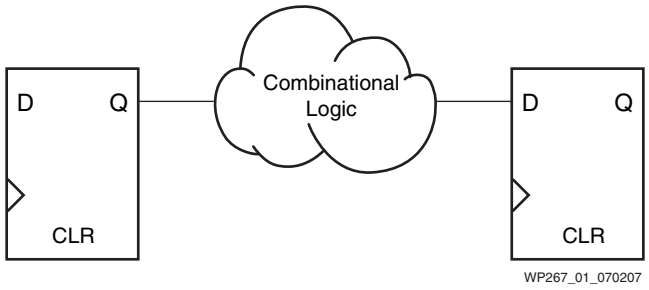

图 1: **标准边界扫描链**

Spartan–3A/3AN/3A DSP 器件是 JTAG 兼容型的，允许配置和读回 FPGA。JTAG 兼容性也意味着 JTAG 引脚无法被禁止。但是，通过采用边界扫描模块，设计者可以进行安全设计，以检测和阻止未授权即使用 JTAG 端口。

边界扫描模块

BSCAN_SPARTAN3A 宏模块（见图 2）允许设计者获取边界扫描信号。通过对该模块进行简单的例示，设计者就可以从 FPGA 内部监控 JTAG 引脚上的活动。

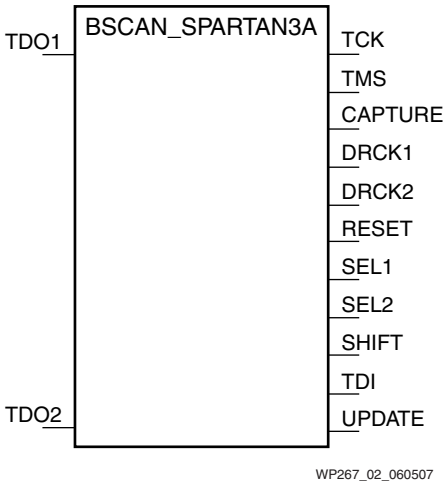

图 2: **BSCAN_SPARTAN3A**

```
-- BSCAN_SPARTAN3A: Boundary Scan primitive for connecting internal
-- logic to JTAG interface
-- Spartan-3A
-- Xilinx HDL Libraries Guide, version 9.1i
BSCAN_SPARTAN3A_inst : BSCAN_SPARTAN3A
port map (
TCK => TCK,
TMS => TMS,
CAPTURE => CAPTURE,      -- CAPTURE output from TAP controller
DRCK1 => DRCK1,          -- Data register output for USER1 functions
DRCK2 => DRCK2,          -- Data register output for USER2 functions
RESET => RESET,          -- Reset output from TAP controller
SEL1 => SEL1,            -- USER1 active output
SEL2 => SEL2,            -- USER2 active output
SHIFT => SHIFT,          -- SHIFT output from TAP controller
TDI => TDI,              -- TDI output from TAP controller
UPDATE => UPDATE,        -- UPDATE output from TAP controller
TDO1 => TDO1,            -- Data input for USER1 function
TDO2 => TDO2,            -- Data input for USER2 function
);
-- End of BSCAN_SPARTAN3A_inst instantiation
```

边界扫描模块如何提高安全性?

如前所述，边界扫描模块可以为从内部监控 JTAG 引脚的活动提供条件。如果检测到端口上的活动，您可以设计逻辑来完全擦除 FPGA 配置或绕过 / 阻止选定的功能。ICAP 可以用于擦除 Spartan–3A/3AN/3A DSP 器件的配置。了解 ICAP 方面的详细解释，请参照 “Spartan–3 系列配置用户指南 (UG332)”。

图 3 给出绕过关键逻辑和功能的实例。该设计将旁路 MUX 整合到由检测逻辑输出控制的关键输入功能中。在正常操作过程中，信号进入逻辑，但是当检测到 JTAG 活动时，信号旁路被断开，并通过逻辑设置一个值。这使得 INTEST 输出对于反向工程内部逻辑来说完全无用。

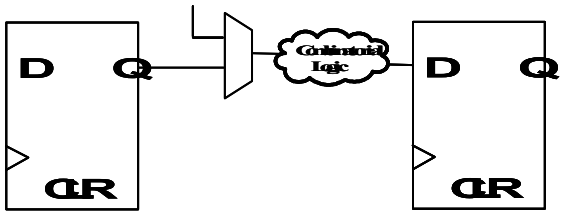

图 3: **用户定义边界扫描链**

在图 4 中，“检测逻辑”可以和门电路一样简单，或者您的应用可能需要更为复杂的逻辑。

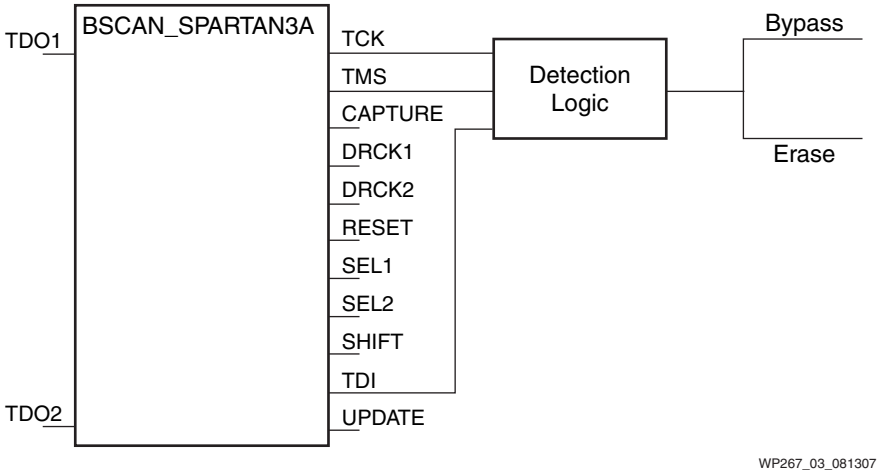

图 4: **检测逻辑**

JTAG 现场更新和安全诊断

在多数情况下，一旦系统或器件部署完毕并开始运转，JTAG 接口就不能被访问或使用了。但是，总有例外的情况。例如，当系统需要现场更新或诊断时，就需要 JTAG 端口。如果为了保护器件免受通过 JTAG 端口进行的未授权访问而实现了检测安全性，这就会阻止授权访问。设计可通过几种方式实现。第一种是设计检测逻辑，它只会激活 INTEST 测试指令，而使 JTAG 在所有其他模式中都正常操作，比如 BYPASS、IDCODE、USERCODE 和 EXTEST。这使得通过现场访问 JTAG 端口进行更新和诊断变得很简单。

对于更为复杂的安全性，检测逻辑可以设计用来监控特定的访问路线或代码序列以允许访问 JTAG 正常操作模式。当现场工作组需要访问 INTEST 指令以对系统功能进行内部测试和验证时，这是很有用的。这可以释放 JTAG 指令，直至诊断测试和升级完成。升级完成后，重启升级后的 FPGA 就可以将检测逻辑复位。对于只进行诊断的系统，可以发布一个代码序列，它将重新开始进行检测逻辑监控。

两种方法都能保证完成现场服务任务，而不会发生通过 JTAG 端口危及安全的情况。如果检测到的序列不正确，就可以用 ICAP 来复位（擦除）。

主动防御逻辑资源要求

Spartan–3 系列器件具有很多内置于芯片中的嵌入式特性和功能。JTAG 状态机和到 ICAP 的接口逻辑就包含在这些嵌入式功能里面。BSCAN_SPARTAN3A 模块不需要逻辑资源，因为该功能是嵌入式的。但是，连至例示的 JTAG 模块的用户逻辑会消耗逻辑和互连资源。该逻辑可以是一个逻辑单元也可以是数十个逻辑单元，这取决于用户逻辑 / 功能的复杂度。

主动防御结论

利用 Spartan–3A/3AN/3A DSP FPGA 进行设计时，只需要利用很少的额外逻辑，就可以通过例示边界扫描模块以及简单的检测逻辑即可检测和提高防反向工程安全性。

比特流验证

本节重点讲述如何阻止篡改配置比特流。对篡改设计感兴趣的人会试图修改原来的设计以便获得未授权服务，盗取敏感数据，或破坏应用。通过在正常操作过程中验证器件配置，就可以检测到被修改的配置，而且设计可以决定如何处理篡改。有很多方法可以实现验证电路。图 5 给出了一个采用 ICAP 和 CRC 的简单实例。

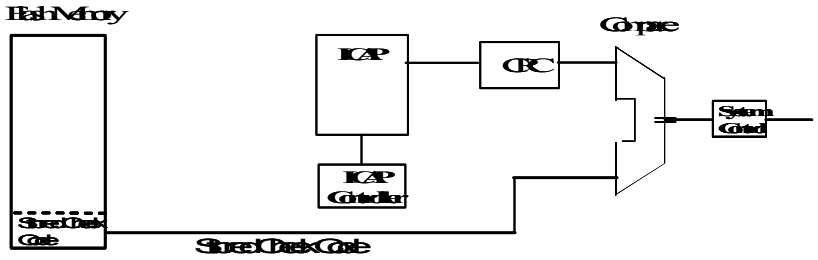

图 5: **比特流验证**

ICAP 模块

ICAP 模块实现了架构和 FPGA 配置控制器之间的接口。该模块基元就像边界扫描模块基元一样，其例示无需额外的逻辑单元，因为这些端口嵌入在了 FPGA 中。要在器件配置完成后读取配置比特流，ICAP 宏必须被例示。ICAP 模块也常用于实现 Spartan–3A/3AN/3A DSP 平台中的多重启动功能。如果 ICAP 被用于实现一种以上的功能，比如多重启动和比特流验证，那么，当连接至 ICAP 时，就需要考虑信号优先级和控制。这可以和多路复用器或更为复杂的仲裁逻辑一样简单。

图 6 给出 ICAP 基元的示意图，后面是 VHDL 例示模板。

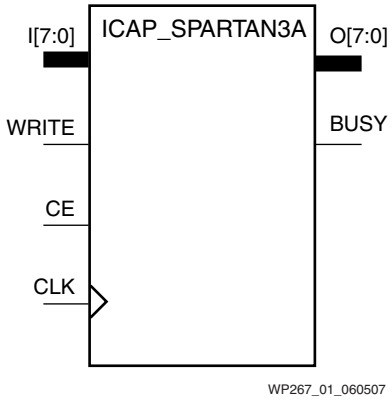

图 6: **ICAP_SPARTAN3A**

```
-- ICAP_SPARTAN3A: Internal Configuration Access Port
--                 Spartan-3A
-- Xilinx HDL Libraries Guide, Version 9.1.3i

ICAP_SPARTAN3A_inst : ICAP_SPARTAN3A
port map (
   BUSY => BUSY,      -- Busy output
   O => O,            -- 8-bit data output
   CE => CE,          -- Clock enable input
   CLK => CLK,        -- Clock input
   I => I,            -- 8-bit data input
   WRITE => WRITE     -- Write input
);

-- End of ICAP_SPARTAN3A_inst instantiation
```

循环冗余校验（CRC）

CRC 是一类校验和，用于检测数据传输和接收中最常出现的误差。它广泛用于蓝牙、以太网、USB、卫星通信，以及 FPGA 的配置中。Xilinx FPGA 具有自检能力，可以在器件加载配置时验证比特流。CRC 是计算的数字结果，并且与生成比特流中的存储值进行比较；如果两个值相等，“Done”引脚变为高电平，表明配置成功。

CRC 算法很简单，但却是一种高度有效的检验数据完整性的方法。散列算法也可用于验证 FPGA 配置。选择 CRC 还是散列算法完全取决于设计者。

简单比特流验证

ICAP 模块用于读取器件配置，然后器件配置被发送至 CRC，CRC 会生成一个有效的结果值。然后，有效值与存储的 CRC 存储值进行比较。在本例中，存储值是一个空配置存储器位置。如果这两个值相同，配置就是正确的。如果这两个值不同，就说明器件已经被篡改，设计者可以决定采取作何反应。一些常用响应如下所示：

- 重新加载配置

  通过使用 ICAP 模块，FPGA 可以被擦除和重新配置。如果主配置已经被篡改，这会导致 FPGA 不断地进行重配置。

- 无功能

  设计完全停止运行。这可以通过采用全局控制信号（如 3 态、门控时钟、触发器时钟使能等）来轻松实现。

根据设计需要也可采用其他响应。

逻辑资源要求

采用嵌入式 ICAP 模块就无需使用 FPGA 内的任何逻辑资源。有多种 CRC 和散列算法可供选择，其中有与几个逻辑单元一样简单的，也有数百个用于实现更为复杂的算法的逻辑单元。

比特流验证结论

对于一些设计来说，保护数据和访问比设计功能更加重要。简单的比特流验证可以协助保护数据、访问和设计功能不被篡改者攻击。

高级数据操作

Device DNA 和存储校验码对于外界来说不是机密；任何人都可以获得这些信息。了解 Device DNA 安全方面的更多信息，请参照 “利用 Spartan–3 系列 FPGA 实现安全解决方案”白皮书 (WP266)。Device DNA 设计级安全的奥密在于 “安全算法”。对于一些设计，安全要求需要超过默认的 57 位 Device DNA 来增强保护，以免受到蛮力攻击。Device DNA 设计具有增加额外位以提高安全性的能力。采用的 Device DNA 位数越多，完成蛮力攻击所需的时间就越长。蛮力攻击是指当克隆者或过度构建者试图通过破解您的安全算法来生成存储校验码。有时，这种攻击所需的时间会非常长，因此在某种程度上不太可能，或不值得进行蛮力攻击。蛮力攻击的总时间是 Device DNA 位数和存储校验码位数的组合。

如图 7 所示，通过采用 Device DNA 的数据操作可提供额外的安全性，以协助阻止蛮力攻击。在本例中，设计为 Device DNA 增加了 64,000 位，存储在 Spartan–3AN 用户 Flash 存储器中。这可以像存储在配置存储器或系统存储器中一样简单。在 Device DNA 后面，在设计中插入一个分类器。分类器其实就是一个多路分配器和一个被解码以便控制多路分配器的选择线路的计数器。多路分配器的第一个输出将数据发送至安全算法，第二个输出将比特位置入位桶。这个简单的电路现在将 Device DNA 关系更改为存储校验码，使得蛮力攻击或反向工程安全算法变得更加困难。

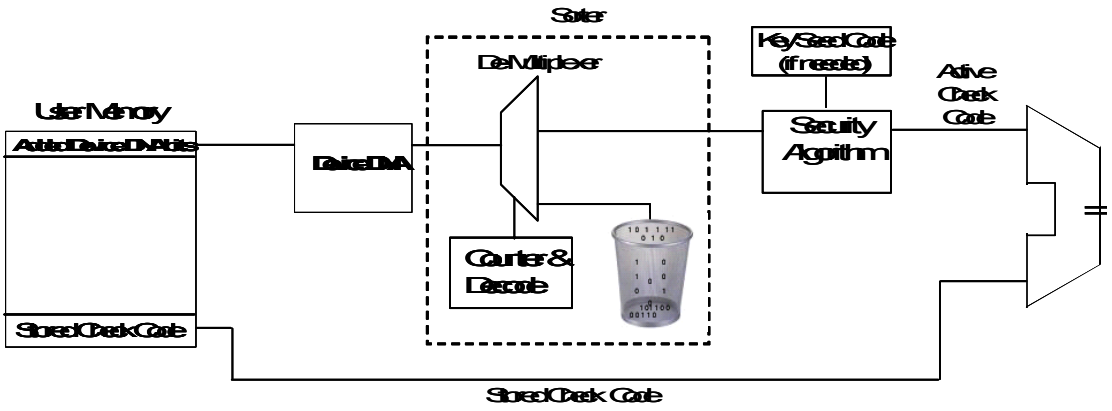

图 7: **Device DNA 的数据操作**

存储校验码和算法控制上的高级数据操作

数据操作技术的进一步扩展可以用于整合存储校验码。图 8 给出一个实例，在这个实例中，数据操作分类器已经被扩展，从而整合了额外的 Device DNA 位和存储校验码。现在，克隆者或过度构建者只看到 Device DNA 被读入 FPGA。这使得克隆者或过度构建者反向工程 Device DNA、存储校验码和垃圾位，然后继续反向工程安全算法变得非常困难。在本例中，添加了第三个多路分配器输出，用以分离存储校验码并将其发送至比较器。

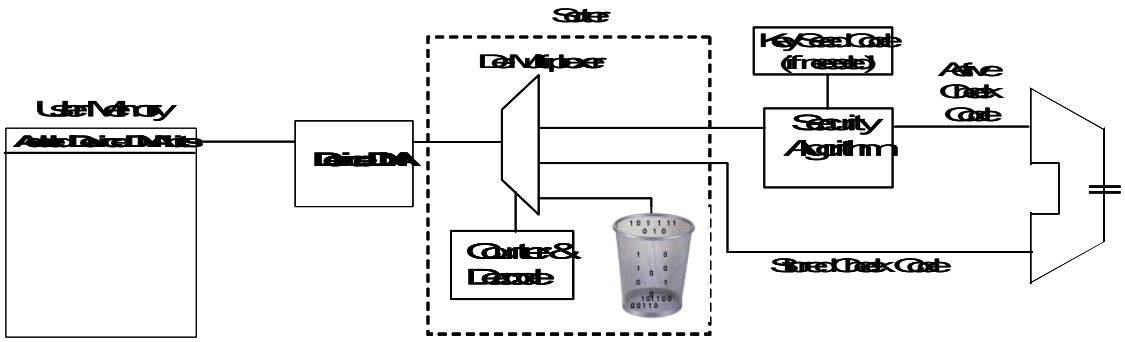

图 8: **存储校验码上的数据操作**

如图 9 所示，通过给多路分配器添加第四个输出并将其直接连至安全算法，可进一步获得该数据操作。基于选择的安全算法，这可以允许设计者改变其种子值、安全密钥，甚至是算法本身，从而形成另一层安全保护，以防克隆或过度构建。通过该数据操作，FPGA 中的硬件设计仍然完全相同，但是安全算法却变了。通过增加安全算法实现的改变可以在制造流程中甚至是现场环境下轻松升级设计安全性。

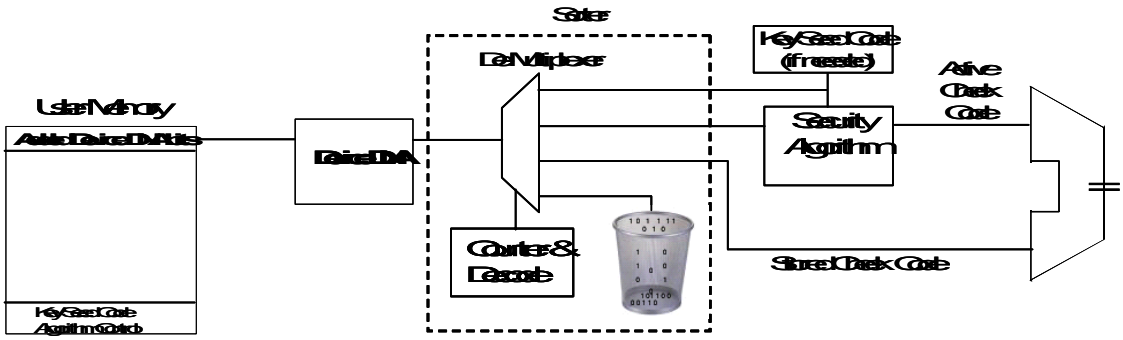

图 9: **给多路分配器添加第四个输出**

逻辑资源要求

数据操作分类器是一个多路分配器和一个被解码以便控制多路分配器的选择线路的计数器，这些线路可以在数十个逻辑单元内实现。

高级数据操作结论

高级数据操作有助于保护 FPGA 设计不被克隆者和过度构建者蛮力攻击，同时还提供了简单而又快速地升级安全性的方法。

结论

本白皮书介绍了多种高级安全机制，设计者可采用这些机制来协助阻止克隆、未授权过度构建、反向工程、以及篡改设计或系统。本文讲述的部分高级方法也是分层技术。该技术整合了不同机制，用以同时降低多种易损性。

修订历史

本技术文档的修订历史如下表所示。

日期	版本	修订
2007 年 8 月 15 日	1.0	Xilinx 最初版本。