

基于弹载存储系统的高速固态数据记录器设计

郭小兵, 李圣昆, 贾兴中

(中北大学 电子测试技术国家重点实验室, 山西 太原 030051)

摘 要: 为了对飞行体发射过程中的参数进行实时存储, 提出了基于 FPGA 的弹载数据记录器电路设计及控制逻辑设计. 应高速数据存储速度的要求, 对 FLASH 在不同编程方式下的写入速度进行了深入分析, 从而革新了 FLASH 控制逻辑设计. 该记录器采用 FPGA 实现接口控制、FLASH 逻辑写入、擦除、无效块校验等, 设计中, FLASH 采用高效的“区”地址管理方法, 从而为满足高速数据存储提出了合理的解决方案. 实践证明, 该记录器对飞行体发射过程中的参数进行了实时、可靠地存储, 满足工程实际需求.

关键词: 固态存储器; FLASH; 现场可编程门阵列; 高速存储; 无效块校验

中图分类号: TP271

文献标识码: A

文章编号: 1000-7180(2013)02-0025-04

The Design of High-Speed Solid-State Data Recorder Based on Onboard Storage System

GUO Xiao-bing, LI Sheng-kun, JIA Xing-zhong

(National Key Laboratory for Electronic Measurement Technology, North University of China, Taiyuan 030051, China)

Abstract: In order to achieve real-time storage of the parameters during the launch of the flight, we give the circuit design and control logic design of onboard data storage based on the FPGA. For the requirements of high-speed data storage, we give an in-depth analysis on the write speed of FLASH in different programming mode via researching the structure of FLASH, which revolutionized the design of FLASH control logic. The recorder implements the interface control, write, erase, invalid block check of FLASH via Field programmable gate array. In this design, we present the reasonable solution to meet the high-speed data storage by using the efficient "zone" address management of FLASH. The practice has proved that the storage achieve the real time, reliable storage of parameters produced in the process of launching of the flight, it has met the needs of the practical engineering.

Key words: solid-state memory; FLASH; FPGA; high-speed storage; invalid block check

1 引言

现场实时完成信息的快速采集与存储, 并回收存储器, 由计算机处理, 是再现被测信息的一种动态测试技术^[1]. 为了提高数据存储的速率, 本文介绍了基于 LVDS 接口的数据存储器的设计与实现方法.

该记录器采用 SAMSUNG 公司的 K9WBG08U1M^[2] 作为记录器的存储介质, 数据输入采用 LVDS 技术, 实现了接收 310Mbps 高速 LVDS 数据流的能力, 通过对 NAND FLASH 控制

逻辑的深入探索, 提出了一种针对 FLASH (K9WBG08U1M) 的高速数据写入方法, 与一般的 FLASH 编程对比, 数据的写入速度有了极大提高, 该设计已成功应用于某型号弹载存储系统^[3].

2 数据记录器电路设计

数据记录器主要完成对采编器输出数据的实时存储, 对输入的高速 LVDS 数据流进行解串^[4], 最终将数据写入 FLASH 存储介质. 数据记录器硬件电路框图如图 1 所示.

收稿日期: 2012-05-18; 修回日期: 2012-06-29

基金项目: 国家自然科学基金项目(60871041)

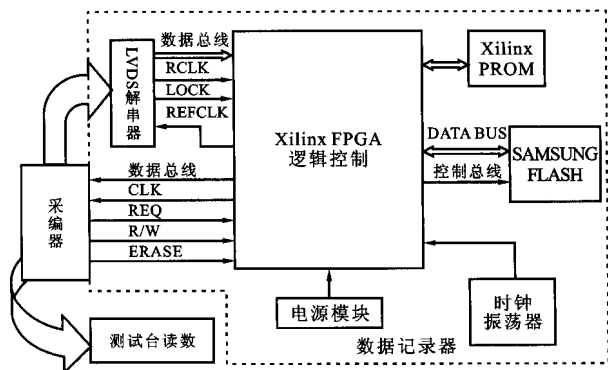


图1 记录器硬件电路图

存储电路主要是对采编器输出的数据进行实时存储,存储器接收数据流的突发速度为31MB/S,而有效数据的平均速度为30MB/S,系统只需以略大于30MB/S的速度完成数据的接收即可。为了将有效数据以零误码率存储到FLASH芯片,通过调用FPGA的IP核,在FPGA中进行异步FIFO的逻辑设计实现数据的缓存,同时控制逻辑以31MB/S的速度将数据从FIFO中读出,并存储到FLASH芯片中,从而实现系统数据接收速度与存储速度的匹配。

3 数据记录器控制逻辑设计

数据记录器的控制逻辑设计主要针对输入的LVDS数据流进行实时高速存储。

3.1 FLASH的区地址划分

从理论的角度讲,Two-Plane Page Program^[1](双平面页编程)和Interleave Two-Plane Page Program^[1](交替双平面页编程)都可以满足高速数据的存储要求;但是在实际操作中可能会出现一些突发情况,为了提高FLASH的存储速度,此次设计采用交叉双平面页编程方式对FLASH进行操作,并采用“区”地址管理方式,区地址分布如图2所示。

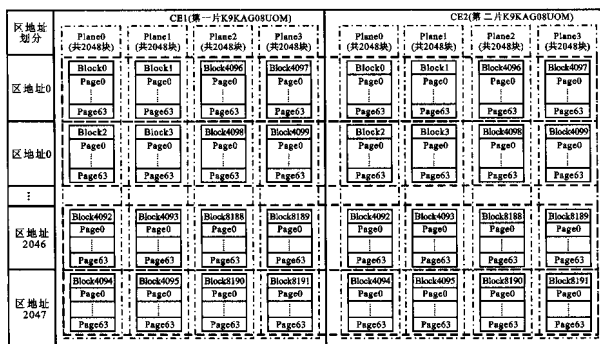


图2 “区”地址分配图

K9WBG08U1M由两片K9KAG08U0M组成,

CE1对应第一片K9KAG08U0M,CE2对应第二片K9KAG08U0M,K9WBG08U1M(页容量为4KB)分为4个阵列进行交替双平面页编程的操作(CE1的Plane0、Plane1为阵列1,Plane2、Plane3为阵列2,CE2的Plane0、Plane1为阵列3,Plane2、Plane3为阵列4),按顺序依次对每一阵列以双平面操作方式进行数据写入,即第一阵列数据写入后,依次进入第二阵列、第三阵列、第四阵列的数据写入。

把各plane中平行的8个block作为一“区”,这样,FLASH的存储空间被分为2048个“区”,每区容量为2MB。对FLASH的操作以“区”为单位,在操作时就可以不必去关注块地址的复杂变化,只需控制CE1/CE2内各阵列的交替以及阵列内两个plane的切换即可。

八个平面按顺序统一管理,定义f_plane[2:0]管理这8个平面;由f_plane(2)控制CE1和CE2的切换。在选定K9KAG08U0M之后,由f_plane(1)控制选定的K9KAG08U0M内阵列的切换,在选定具体的某一阵列后,由f_plane(0)控制偶数块和奇数块之间的切换。f_plane=“000”时选中CE1的阵列1中的偶数块;“001”时选中CE1的阵列1中的奇数块;“010”时选中CE1的阵列2中的偶数块;“011”时选中CE1的阵列2中的奇数块;“100”时选中CE2的阵列3中的偶数块;“101”时选中CE2的阵列3中的奇数块;“110”时选中CE2的阵列4中的偶数块;“111”时选中CE2的阵列4中的奇数块;同时由区地址控制操作哪个区。

定义区地址变量f_countqu,无效区校验时5个周期的地址写入代码中,地址的设置:

```
1st: flashdo <= "00000000";
2st: flashdo <= "00010000";
3st: flashdo <= "00000&p_switch";
flashdo <= "f_countqu(0) & f_plane(0)";
4st: flashdo <= "f_countqu(8 downto 1)";
5st: flashdo <= "00000&f_plane(1) & f_countqu(10 downto 9)";
```

另外,CE1和CE2的切换由f_plane(2)控制的代码如下:

```
CE1 <= f_plane(2);
CE2 <= not f_plane(2);
```

本设计中高效的地址管理方式——“区”地址管理方式,不再关注对复杂块地址的控制,而改为控制“区地址”和plane地址,从而避免了块地址跳变的麻烦,简化了逻辑,提高了速度。

3.2 不同编程方式下 FLASH 写入速度的分析

当对 FLASH 中的无效区识别完成后,就可以进入正常的擦除、写入、读数的状态;而影响数据存储速度除了与 FLASH 无效块有关,同时与 FLASH 的页编程方式有关^[5]. 一般比较常用的 FLASH 页编程方式有三种:Page Program、Interleave page program(交替页编程^[5])、Interleave Two-Plane Page Program(交叉双平面页编程^[5]). 以 K9WBG08U1M 的其中一片 K9KAG08U0M 为例(FLASH 峰值写入速度为 31 MB/s),对各种编程方式下芯片的平均写入速度进行深入分析:

(1)普通页编程(page program),每次操作的块个数为 1,平均写入速度为:

$$\frac{4\text{KB}}{700\mu\text{s} + \frac{4\text{KB}}{31\text{MB/s}}} = 5.7\text{MB/s}$$

(2)交替页编程(Interleave page program),每次操作的块个数为 2,平均写入速度为:

$$\frac{4\text{KB} * 2}{700\mu\text{s} + \frac{4\text{KB}}{31\text{MB/s}}} = 11.4\text{MB/s}$$

(3)交替双平面页编(Interleave two-plane page program),每次操作的块个数为 4,平均写入速度为:

$$\frac{4\text{KB} * 4}{700\mu\text{s} + \frac{4\text{KB} * 2}{31\text{MB/s}}} = 22.8\text{MB/s}$$

综上所述可知,交替双平面页编程方式与普通页编程方式、交替页编程方式相比,极大的提高了 FLASH 芯片的平均写入速度.

3.3 交替双平面页编程的时序

交替双平面页编程过程中,经实验测试,完成对一个阵列内两个 plane 的装载操作需要的时间为 210~350 μs ,完成该阵列两个 plane 的自动编程操作需要 250~700 μs . 为了最大限度的“省略”自动编程时间,同时结合 FLASH 内部结构,在内部两块 K9KAG08U0M 交替的同时,加入 K9KAG08U0M 内阵列的交替. 在第一片 K9KAG08U0M 内的阵列 1 装载完毕进入自动编程的同时,进行对阵列 2 的装载,其装载完毕后进入对第二片 K9KAG08U0M 的操作……即依次进行对 4 个 plane 阵列的装载和自动编程,并以此顺序循环. 这样,从某阵列开始装载到下一次装载,耗时最小为 $4 * 210\mu\text{s} = 840\mu\text{s} > 700\mu\text{s}$,即再次对该阵列装载时,其已经完成对已装载数据的自动编程,省略了等待时间,做到了时间利

用的最大化.

在对某一阵列操作时,先写命令 80h 或 81h,然后将地址和数据先后写入数据总线,最后写入命令 11h 或 10h 启动编程操作. 其中,命令字 80h 和 11h 控制阵列内第一个 Plane 的装载过程,命令字 81h 和 10h 用于控制另一个 Plane. 在第一个 Plane 数据装载至第一个 Plane 的页寄存器后,输入的不是自动编程的确认命令 10h,而是假确认命令 11H,FLASH 芯片进入虚拟编程(tDBSY),该过程只是短暂停顿,其典型时长只有 500ns,最长 1 μs . 加载完第二个 Plane 数据后输入命令 10h,该命令开启该阵列两个 plane 的同步编程. 交替双平面页编程阵列内装载、编程时序图如图 3 所示.

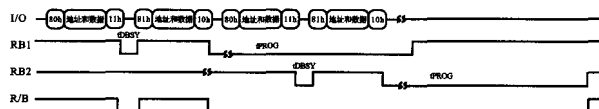


图3 阵列内装载、编程时序图

图 3 中, RB1 和 RB2 分别为 FLASH 芯片 K9WBG08U1M 中 K9KAG08U0M 内两个阵列的忙闲信号, R/B 为 K9KAG08U0M 的忙闲信号. 地址为数据存储的初始位置,数据将从该地址开始顺序存入芯片.

3.4 无效块地址管理

对于 FLASH 中无效块的识别通常有两种方法:方法一为:在系统工作后,对每块进行操作之前判断当前块的第一页和第二页的列地址为 4096 单元的内容是否全部为 FF,块地址自加后,重复之前的判断操作,即对每一块操作之前,必须读取当前块的第一页和第二页的列地址为 4096 单元中的数据;第二种方法:系统上电后,开始判断 FLASH 中的所有块,当检测到无效块时将无效块的块地址存入一个特定的缓存中,创建无效块地址列表,在对 FLASH 进行擦除、读、写操作之前,只需将当前操作的块地址与事先建立好的无效块地址列表按顺序一一进行比较判断. 如果当前块地址与无效块地址列表中读取的无效块地址相同,则认为当前块为无效块,并且跳到下一块,读取缓存列表中下一个无效块地址值与其比较;否则,认为当前块为有效块,可以对其进行正常操作.

在实际应用中,若时钟晶振选用 62 MHz,则检验完 FLASH 中一个块地址是否为无效块所用的时间最大约为 40 μs . 若采用方法一进行操作,则在每次操作前至少需要耗时 40 μs 进行无效块的校验,才

能对该块进行操作;如果当前操作的块为无效块,则跳过该块,重新进行判断,从而进一步增加无效块检验的时间,降低了 FLASH 的数据存储的速度.若采用方法二进行操作,则在每次操作前只需读出缓存中的无效块地址值与当前无效块地址进行比较即可,一次判断大约最多只需要 67 ns(读取 RAM 缓存中的无效块地址时间).

虽然第二种方法相对第一种方法其速度有很大提高,但不适于“区”地址管理方式的“轮循”编程方式.本设计中采用了一种特殊的无效“区”地址管理方式:调用 FPGA 内部的 IP 核,构建 RAM(容量为 2KB).将无效“区”地址 $f_countqu$ 写入到该 RAM 列表中.区操作的流程图如图 4 所示.

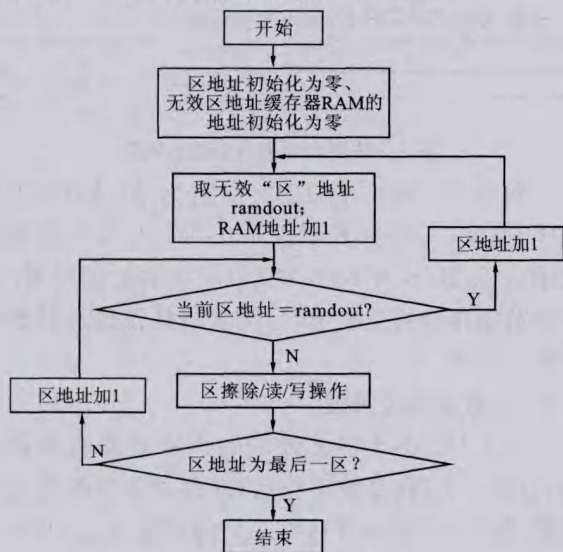


图4 “区”操作流程图

在区地址分布图中,当 CE1 中的 block($2 * n + 0$)、block($2 * n + 1$)、block($2 * n + 4096$)、block($2 * n + 4097$)和 CE2 中的 block($2 * n + 0$)、block($2 * n + 1$)、block($2 * n + 4096$)、block($2 * n + 4097$)的任何一块为无效块时,则这八块都认为是无效块,直接跳过这八块,并把这八块对应的无效“区”地址 $f_countqu$ 写入事先建立好的 RAM 列表中.按照该方式依次记录各无效“区”号,写入 RAM 中.在对“区”操作前,只需将其“区”号与 RAM 中的值对比即可,无需重新判断无效块.

在无效块扫描完毕,开始“区”操作时(待操作区从区 0 开始),先读取 RAM 中的第一个值,如果待操作“区”号与该值不相等,则可对该“区”进行操作,同时将区号加 1;反之,则跳过该区(区号加 1),读取 RAM 中的下一个值,重复以上操作.

这种方式实际上是以“浪费”FLASH 的存储空

间为代价,实现了对无效块的高效管理.经抽样检测,该 FLASH 芯片的无效块数一般在 20~50 块,采用该无效块管理方式“浪费”的存储空间最多为 $50 * 8 * 256K = 100MB$,对于设计是没有影响的.

4 记录器实测

测试试验的目的是验证记录器在数据实时存储中的可行性与可靠性^[6].试验前,采编器采集标准的模拟弹上信源(正弦波、方波、直流量、锯齿波……依次循环),并将其存入记录器,测试弹发射试验后将事先存储在存储器中的波形数据回收,再用上位机软件对其进行数据波形绘制,其对应的数据波形如图 5 所示.从图中可以看出,其回收的数据波形未出现异常,并与试验前的数据波形一致.由此可知,该数据记录器可靠的完成了数据的实时存储.

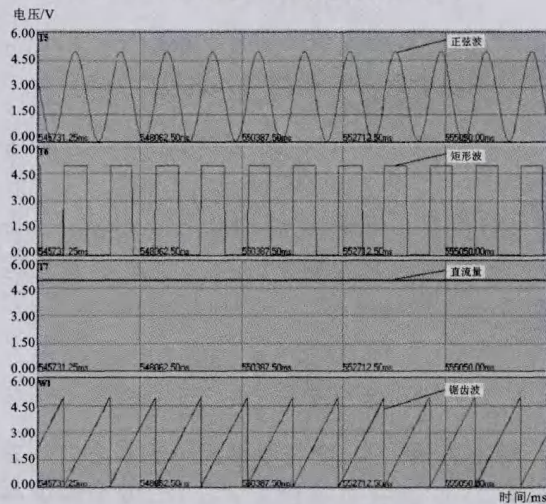


图5 发射试验后回收的数据波形

5 结束语

本文给出了高速数据记录器的电路设计和控制逻辑设计.详细地介绍了数据记录器的硬件电路设计以及数据的高速存储逻辑设计,实现了 FLASH 的高效区地址管理方式,该记录器与采用一般的编程方式实现的记录器相比,极大的提高了数据存储的速度,目前,本设计已成功应用于某型弹载数据存储系统.

参考文献:

- [1] 赵艳利,甄国勇,文丰.多路数据采集存储测试系统设计与应用[J].电子设计工程,2008(10):17.
- [2] 王洋,杨国武,郑德生,等. NAND Flash 仿真模块的一种验证方法[J].微电子学与计算机,2011,28(5):32-35.

(下转第 32 页)

```

<place id="pi"><... /></place>其它
库所
<transition id="t1">
<name><text>IF</text></name>IF
迁移
</transition>
<transition id="ti"><... /></transition>
>其它迁移
<arc id="a1" source="p1" target="t1">
</arc>弧的描述
<arc id="ai" source="pj" target="tk">
</arc>
</page></net></pnml>

```

限于篇幅,未给出指令流水线细节和可视显示部分 PNML 描述。

6 结束语

处理器设计的核心问题之一是体系结构中的流水线状态机的描述与综合。Petri 网作为一种系统建模工具,特别适合描述系统的控制流、并发和异步特性,与状态机行为模型相比更具优势,并且可被逻辑综合与电路实现。

高级 CPN 模型中 token 可以带有信息,克服了传统 Petri 网中仅仅将其看作一般初始化的不足。在描述多种相关情形时,通过定义颜色集对进行模型简化和规整,加强了 Petri 网的建模能力。

PNML 描述语言就是一种极具潜力的设计方

法,它规范了各种 Petri 网的语义和行为,可以预期验证系统功能的正确性,还能作为综合工具的输入以及系统技术文档标准,并且可以通过第三方工具扩展其可执行描述功能。

参考文献:

- [1] Gonzalez R E, Xtensa. A configurable and extensible processor[J]. IEEE Micro, 2000, 20(2):60-70.
- [2] Segre Leef. A methodology for virtual hardware/software integration[M]. Mentor,1998.
- [3] Itoh M, Higaki S, Sato J, et al. PEAS2-III: an ASIP design environmen[J]. Computer Design, 2000; 430 - 436.
- [4] Oliver Schliebusch. Architecture implementation using the machine description language LISA[C]//Proceedings of the 15th International Conference on VLSI Design. India;Bangalore,2002;239 - 244.
- [5] Henk Corporaal. Microprocessor Architectures: from VLIW to SDTA[M]. New York;John Wiley, 1998.
- [6] 朱勇. 嵌入式 RAID 数据通道的逻辑描述[J]. 小型微型计算机系统,2004(2):254-256.
- [7] 朱勇. 嵌入式高速通道逻辑描述[D]. 武汉:华中科技大学,2007.
- [8] A. Cilio. Architecture definition file[D]. Finland: Tampere University of Technology,2007.
- [9] Billington J. The petri net markup language: concepts, technology, and tools[C]// 24th International Conference, LNCS 2679. Springer, 2003; 483 - 505.

(上接第 28 页)

- [3] Samsung Electronics. K9XXG08XXM Flash Miemory [EB/OL]. [2011-05-06]. www.samsung.com; 18-43.
- [4] National Semiconductor. DS92LV1224 30-66MHz 10 Bit Bus LVDS Deserializer[EB/OL]. [2009-10-06]. www.national.com. 2-3.
- [5] 李佳琦,李健,党相甫,等. 基于固态硬盘的闪存阵列并行结构设计[J]. 微电子学与计算机,2010, 27(4): 85-88.
- [6] 郭小兵,李圣昆,任勇峰. 基于双备份多路数据采集存储系统的设计与实践[J]. 电子技术应用,2012,38(2):

67-69.

作者简介:



郭小兵 男,(1986-),硕士研究生.研究方向为微系统集成、嵌入式系统、高速数据采集、测试技术。