(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 104572326 A
(43)申请公布日 2015.04.29

(21)申请号 201410789605.3

(22)申请日 2014.12.18

(71)申请人 北京时代民芯科技有限公司
地址 100076 北京市丰台区东高地四营门北路2号
申请人 北京微电子技术研究所

(72)发明人 兰利东 陆振林 赵元富 王建永 焦烨 李璟 刘薇 王猛 李楠

(74)专利代理机构 中国航天科技专利中心 11009
代理人 陈鹏

(51)Int.Cl.
G06F 11/07(2006.01)

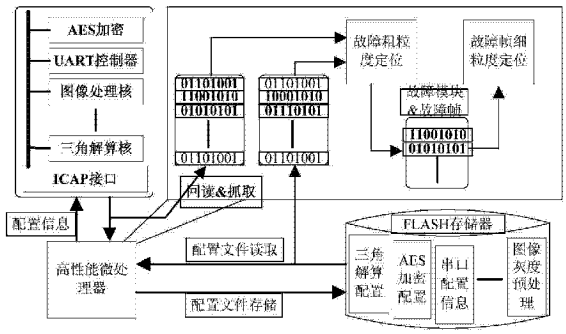
权利要求书3页 说明书10页 附图1页

(54)发明名称

一种基于回读自重构的SoPC芯片容错方法

(57)摘要

一种基于回读自重构的SoPC芯片容错方法，针对传统的辐射加固的FPGA在太空中每天会发生多次粒子翻转的问题，提供了一种在不增加SoPC芯片面积的前提下，基于回读自重构的方式实现芯片的自主故障检测及故障修复的方法。本发明方法首先读取FPGA配置存储器中的配置数据和Flash中存储的原始配置数据，然后将两者逐位进行比较，通过比较文件格式上的差异能够验证回读的配置数据是否发生了故障并定位故障，最后根据原始配置文件将故障纠正。本发明方法在不增加外围检测设备和检测电路的情况下完成了故障检测、故障判读、故障修复，提高了SoPC芯片在外太空环境应用中的可靠性，推动了SoPC芯片的发展。



CN 104572326 A

1.一种基于回读自重构的SoPC芯片容错方法,其特征在于包括如下步骤:

(1)将FPGA的模式选择引脚M0、M1、M2均连接到FPGA的GND端;

(2)将SPARC V8处理器的GPIO-PIO48引脚连接到FPGA的IO_D7引脚,GPIO-PIO49引脚连接到FPGA的IO_D6引脚,GPIO-PIO50引脚连接到FPGA的IO_D5引脚,GPIO-PIO51引脚连接到FPGA的IO_D4引脚,GPIO-PIO52引脚连接到FPGA的IO_D3引脚,GPIO-PIO53引脚连接到FPGA的IO_D2引脚,GPIO-PIO54引脚连接到FPGA的IO_D1引脚,GPIO-PIO55引脚连接到FPGA的IO_D0引脚,GPIO-PIO62连接到FPGA的IO_WRITE引脚,GPIO-PIO63引脚连接到FPGA的IO_CS引脚,GPIO-PIO61引脚连接到FPGA的GCLK引脚,GPIO-PIO56引脚连接到FPGA的INIT引脚,GPIO-PIO58引脚连接到FPGA的DONE引脚,GPIO-PIO60引脚连接到FPGA的IO_DOUT_BUSY引脚,GPIO-PIO57引脚连接到FPGA的PROGRAM引脚;

(3)将SPARC V8处理器地址线Ai接到FLASH的地址线PAi-1,i = 1,2,3,,,22,数据线Dj,j = 16,17,18,,,31,连接到FLASH的数据线DQg,g = 0,1,2,,,15,写控制信号端连接到FLASH的WE端,复位控制信号端连接到FLASH的PRESET端,片选控制信号端连接到FLASH的CE端,读控制信号端连接到FLASH的OE端;

(4)将SPARC V8处理器地址线Ai接到SDRAM的地址线PAi-1,数据线Dj,连接到SDRAM的数据线DQk,k = 0,1,2,,,31,写控制信号端连接到SDRAM的SDWEN端,复位控制信号端连接到FLASH的SDRASN端,片选控制信号端连接到SDRAM的SDCS0端,字节控制信号端BE0连接到SDDQM0,字节控制信号端BE1连接到SDDQM1,字节控制信号端BE2连接到SDDQM2,字节控制信号端BE3连接到SDDQM3;

(5)设置SPARC V8处理器GPIO-PIO63引脚为输出,向SPARC V8处理器GPIO-PIO63的数据寄存器写入“0”,经过1个时钟周期后,设置SPARC V8处理器GPIO-PIO62脚为输出,并向SPARC V8处理器GPIO-PIO62的数据寄存器写入“0”;

(6)设置SPARC V8处理器的GPIO-PIO61引脚为输出,向SPARC V8处理器的GPIO-PIO61的数据寄存器写入“1”,下一个时钟周期向SPARC V8处理器的GPIO-PIO61的数据寄存器写入“0”,向SPARC V8处理器的GPIO-PIO61的数据寄存器写入“1”和“0”交替进行,并同时执行步骤(7)-步骤(8);

(7)设置SPARC V8处理器的GPIO-PIO63引脚为输出,向SPARC V8处理器GPIO-PIO63的数据寄存器写入“0”,设置SPARC V8处理器的GPIO-PIO62脚为输出,并向SPARC V8处理器GPIO-PIO62的数据寄存器写入“0”,设置SPARC V8处理器的GPIO-PIO57引脚为输出,并向SPARC V8处理器的GPIO-PIO57的数据寄存器写入“1”;

(8)根据Xilinx手册,生成回读命令数组,从回读命令数组中依次取出数据,当CCLK信号为高电平时,以2进制的格式,从高到低写入SPARC V8处理器的GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55,直至回读命令数组被全部遍历,当回读命令数组被全部遍历时,转入步骤(9);所述回读命令数组包括同步字、写入到FAR寄存器、起始帧地址、写入到CMD寄存器、包数据RCFG、从FDRO寄存器中读出、数据字;

(9)设置SPARC V8处理器GPIO-PIO62引脚与GPIO-PIO63引脚均为输出,并分别向SPARC V8处理器GPIO-PIO62的数据寄存器与GPIO-PIO63的数据寄存器写入“1”,设置SPARC V8处理器GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、

GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 引脚为输入，再次设置 SPARC V8 处理器 GPIO-PIO62 引脚与 GPIO-PIO63 引脚均为输出，并分别向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器与 GPIO-PIO63 的数据寄存器写入“1”，设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”；

(10) 设置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”；

(11) 设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”和“0”交替进行，并同时执行步骤 (12)- 步骤 (13)；

(12) 设置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，设置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”；

(13) 当 CCLK 信号为高电平时，以 2 进制的格式从 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 读取数据，直至 FPGA 中的数据全部读出，得到回读数据；所述回读数据为多个 CLB 数据帧组成的数据帧数组；

(14) 设置 SPARC V8 处理器 GPIO-PIO62 引脚与 GPIO-PIO63 引脚均为输出，并分别向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器与 GPIO-PIO63 的数据寄存器写入“1”；

(15) 将回读得到的数据帧数组与存储在 FLASH 中的原始有效配置数据进行比较，如果数据出现不一致，则根据不一致的数据所在数据帧的帧标识得到故障帧的行、列坐标信息，如果数据没有出现不一致，则 FPGA 数据没有发生粒子翻转；

(16) 在 ISE10.1 开发环境任意生成的 .bit 文件并送至 FLASH 中，对 SPARC V8 处理器、FLASH、FPGA 进行上电，SPARC V8 处理器从 FLASH 中读取 .bit 文件，按照 .bit 文件格式和 Virtex 芯片的配置格式生成配置信息数组；

(17) 设置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，向 GPIO-PIO57 引脚的数据寄存器中写入“1”，然后向 SPARC V8 处理器的 GPIO-PIO57 引脚的数据寄存器中写入“0”，设置 SPARC V8 处理器的 GPIO-PIO56 引脚为输入，监测 FPGA 的 INIT 引脚的电压变化；

(18) 如果 INIT 引脚由低电平变为高电平，则转入步骤 (19)；如果 FPGA 的 INIT 引脚为高电平，则重复步骤 (16)- 步骤 (17)，直至 FPGA 的 INIT 引脚出现由低电平向高电平的跳变后转步骤 (19)；

(19) 设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，然后向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，一个时钟周期后设置 SPARC V8 处理器 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO 的数据寄存器写入“0”；

(20) 设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的

数据寄存器写入“0”，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”和“0”交替进行，并同时执行步骤 (21)- 步骤 (22)；

(21) 设置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”；

(22) 从配置信息数组中依次取出数据，在 FPGA 的 GCLK 信号为高电平时，以 2 进制的格式从高到低写入 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55，同时持续监控 FPGA 的 IO_DOUT_BUSY 引脚，如果 IO_DOUT_BUSY 引脚为高电平，则持续向 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 的数据寄存器中写入当前配置数据，直至 IO_DOUT_BUSY 信号输出为低电平，如果 IO_DOUT_BUSY 为低电平，则转入步骤 (23)；

(23) 设置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，然后设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“1”；

(24) 配置 SPARC V8 处理器 GPIO-PIO58 引脚为输入，持续检测 FPGA 的 DONE 控制信号的输出，如果 DONE 控制信号电平为高，则软配置电路完成，如果 DONE 控制信号电平不为高，则继续等待，直至 DONE 信号为高，当等待时间超出设定的阈值时，则重复执行 (16) 至 (23) 直至 DONE 信号为高。

2. 根据权利要求 1 所述的一种基于回读自重构的 SoPC 芯片容错方法，其特征在于：所述的同步字为 0xAA995566h、写入到 FAR 寄存器为 0x30002001h、起始帧地址为 0x00000000h、写入到 CMD 寄存器为 0x30008001h、包数据 RCFG 为 0x00000000h、从 FDRO 寄存器中读出为 0x28006000h、数据字为 0x4800cb07h。

一种基于回读自重构的 SoPC 芯片容错方法

技术领域

[0001] 本发明涉及一种国产 SoPC(Programing System on Chip) 芯片 BM3109 的智能化自主容错和故障修复，特别是一种基于回读自重构的 SoPC 芯片容错方法，使得 SoPC 芯片能够在外太空环境下，具有自主故障检测和修复的能力。

背景技术

[0002] 传统的辐射加固的 FPGA 在太空中每天会发生多次粒子翻转 (SEU)，比如 LEO 轨道的 XQVR300 型 FPGA 每天平均翻转 2.05 次，而 98 度倾斜轨道的 XQR4036XL 在太阳耀斑异常时每天翻转多达 148.5 次。为了适应恶劣的空间环境，传统的星载电子设备一般采用冗余的方法来提高系统的可靠性，比如双机备份、三机备份等多种冗余方案，然而冗余方案大都针对已知的故障模式设计的，对于复杂的和未知的故障处理则缺乏有效的办法。另外即使采用三模冗余 TMR 的机制，最多也只能容纳一个故障，当存在两个或者两个以上的故障时，会因为数据仲裁器无法给出正确的仲裁结果而导致错误的发生。三模冗余的方式只能过滤故障，并不能对出现的故障进行修复。同时，片上卫星自身体积极小，采用冗余的设计方案必将导致体积的增加，与片上卫星的设计理念相悖。

发明内容

[0003] 本发明解决的技术问题是：克服现有技术的不足，提供了一种在不增加 SoPC 芯片面积的前提下，基于回读自重构的方式实现芯片的自主故障检测及故障修复的方法，提高了 SoPC 芯片在外太空环境应用中的可靠性。

[0004] 本发明的技术解决方案是：一种基于回读自重构的 SoPC 芯片容错方法，包括如下步骤：

[0005] (1) 将 FPGA 的模式选择引脚 M0、M1、M2 均连接到 FPGA 的 GND 端；

[0006] (2) 将 SPARC V8 处理器的 GPIO-PIO48 引脚连接到 FPGA 的 IO_D7 引脚，GPIO-PIO49 引脚连接到 FPGA 的 IO_D6 引脚，GPIO-PIO50 引脚连接到 FPGA 的 IO_D5 引脚，GPIO-PIO51 引脚连接到 FPGA 的 IO_D4 引脚，GPIO-PIO52 引脚连接到 FPGA 的 IO_D3 引脚，GPIO-PIO53 引脚连接到 FPGA 的 IO_D2 引脚，GPIO-PIO54 引脚连接到 FPGA 的 IO_D1 引脚，GPIO-PIO55 引脚连接到 FPGA 的 IO_D0 引脚，GPIO-PIO62 连接到 FPGA 的 IO_WRITE 引脚，GPIO-PIO63 引脚连接到 FPGA 的 IO_CS 引脚，GPIO-PIO61 引脚连接到 FPGA 的 GCLK 引脚，GPIO-PIO56 引脚连接到 FPGA 的 INIT 引脚，GPIO-PIO58 引脚连接到 FPGA 的 DONE 引脚，GPIO-PIO60 引脚连接到 FPGA 的 IO_DOUT_BUSY 引脚，GPIO-PIO57 引脚连接到 FPGA 的 PROGRAM 引脚；

[0007] (3) 将 SPARC V8 处理器地址线 Ai 接到 FLASH 的地址线 PAi-1，i ＝ 1,2,3,,,22，数据线 Dj，j ＝ 16,17,18,,,31，连接到 FLASH 的数据线 DQg，g ＝ 0,1,2,,,15，写控制信号端连接到 FLASH 的 WE 端，复位控制信号端连接到 FLASH 的 PRESET 端，片选控制信号端连接到 FLASH 的 CE 端，读控制信号端连接到 FLASH 的 OE 端；

[0008] (4) 将 SPARC V8 处理器地址线 Ai 接到 SDRAM 的地址线 PAi-1，数据线 Dj，连接到

SDRAM 的数据线 DQk，k ＝ 0,1,2,,,31，写控制信号端连接到 SDRAM 的 SDWEN 端，复位控制信号端连接到 FLASH 的 SDRASN 端，片选控制信号端连接到 SDRAM 的 SDCS0 端，字节控制信号端 BE0 连接到 SDDQM0，字节控制信号端 BE1 连接到 SDDQM1，字节控制信号端 BE2 连接到 SDDQM2，字节控制信号端 BE3 连接到 SDDQM3 ；

[0009] (5) 设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，经过 1 个时钟周期后，设置 SPARC V8 处理器 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”；

[0010] (6) 设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”和“0”交替进行，并同时执行步骤 (7)- 步骤 (8) ；

[0011] (7) 设置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”；

[0012] (8) 根据 Xilinx 手册，生成回读命令数组，从回读命令数组中依次取出数据，当 CCLK 信号为高电平时，以 2 进制的格式，从高到低写入 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55，直至回读命令数组被全部遍历，当回读命令数组被全部遍历时，转入步骤 (9) ；所述回读命令数组包括同步字、写入到 FAR 寄存器、起始帧地址、写入到 CMD 寄存器、包数据 RCFG、从 FDRO 寄存器中读出、数据字 ；

[0013] (9) 设置 SPARC V8 处理器 GPIO-PIO62 引脚与 GPIO-PIO63 引脚均为输出，并分别向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器与 GPIO-PIO63 的数据寄存器写入“1”，设置 SPARC V8 处理器 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 引脚为输入，再次设置 SPARC V8 处理器 GPIO-PIO62 引脚与 GPIO-PIO63 引脚均为输出，并分别向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器与 GPIO-PIO63 的数据寄存器写入“1”，设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”；

[0014] (10) 设置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”；

[0015] (11) 设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”和“0”交替进行，并同时执行步骤 (12)- 步骤 (13) ；

[0016] (12) 设置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，设置 SPARC V8 处理器的 GPIO-PIO57

引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”；

[0017] (13) 当CCLK信号为高电平时，以2进制的格式从SPARC V8处理器的GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55读取数据，直至FPGA中的数据全部读出，得到回读数据；所述回读数据为多个CLB数据帧组成的数据帧数组；

[0018] (14) 设置SPARC V8处理器GPIO-PIO62引脚与GPIO-PIO63引脚均为输出，并分别向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器与 GPIO-PIO63 的数据寄存器写入“1”；

[0019] (15) 将回读得到的数据帧数组与存储在 FLASH 中的原始有效配置数据进行比较，如果数据出现不一致，则根据不一致的数据所在数据帧的帧标识得到故障帧的行、列坐标信息，如果数据没有出现不一致，则 FPGA 数据没有发生粒子翻转；

[0020] (16) 在 ISE10.1 开发环境任意生成的 .bit 文件并送至 FLASH 中，对 SPARC V8 处理器、FLASH、FPGA 进行上电，SPARC V8 处理器从 FLASH 中读取 .bit 文件，按照 .bit 文件格式和 Virtex 芯片的配置格式生成配置信息数组；

[0021] (17) 设置SPARC V8处理器的GPIO-PIO57引脚为输出，向GPIO-PIO57引脚的数据寄存器中写入“1”，然后向 SPARC V8 处理器的 GPIO-PIO57 引脚的数据寄存器中写入“0”，设置 SPARC V8 处理器的 GPIO-PIO56 引脚为输入，监测 FPGA 的 INIT 引脚的电压变化；

[0022] (18) 如果 INIT 引脚由低电平变为高电平，则转入步骤 (19)；如果 FPGA 的 INIT 引脚为高电平，则重复步骤 (16)- 步骤 (17)，直至 FPGA 的 INIT 引脚出现由低电平向高电平的跳变后转步骤 (19)；

[0023] (19) 设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，然后向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，一个时钟周期后设置 SPARC V8 处理器 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO 的数据寄存器写入“0”；

[0024] (20) 设置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”和“0”交替进行，并同时执行步骤 (21)- 步骤 (22)；

[0025] (21) 设置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO 的数据寄存器写入“0”，设置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”；

[0026] (22) 从配置信息数组中依次取出数据，在 FPGA 的 GCLK 信号为高电平时，以 2 进制的格式从高到低写入 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55，同时持续监控 FPGA 的 IO_DOUT_BUSY 引脚，如果 IO_DOUT_BUSY 引脚为高电平，则持续向 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 的数据寄存器中写入当前配置数据，直至 IO_DOUT_BUSY 信号输出为低电平，如果 IO_DOUT_BUSY 为低电平，则转入步骤 (23)；

[0027] (23) 设置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，然后设置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向

SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“1”；

[0028] (24) 配置 SPARC V8 处理器 GPIO-PIO58 引脚为输入，持续检测 FPGA 的 DONE 控制信号的输出，如果 DONE 控制信号电平为高，则软配置电路完成，如果 DONE 控制信号电平不为高，则继续等待，直至 DONE 信号为高，当等待时间超出设定的阈值时，则重复执行 (16) 至 (23) 直至 DONE 信号为高。

[0029] 所述的同步字为 0xAA99 5566h、写入到 FAR 寄存器为 0x3000 2001h、起始帧地址为 0x0000 0000h、写入到 CMD 寄存器为 0x3000 8001h、包数据 RCFG 为 0x0000 0000h、从 FDRO 寄存器中读出为 0x2800 6000h、数据字为 0x4800 cb07h。

[0030] 本发明与现有技术相比的优点在于：

[0031] (1) 本发明方法使 SoPC 芯片具有了自主在轨故障监测和修复的能力。采用传统 SoPC 的设计及实现方法，当芯片随航天器进入外太空环境后，只能采取冗余手段进行故障的过滤，在出现故障后，没有有效的手段进行修复，本发明方法可以使 SoPC 芯片自主实现故障的监测及恢复的能力；

[0032] (2) 本发明方法更加智能化，传统的故障恢复手段常采用专用配置芯片在 SOPC 芯片运行一段时间后，对 SoPC 芯片执行刷新操作来实现故障的屏蔽，整个执行过程中都是按照既定时间进行盲目刷新操作，即使 SOPC 中并不存在错误或故障。这将导致系统频繁的进行数据保护和逻辑重构，增加了系统的可靠性和实时性。而本发明方法是在识别到故障后再进行故障修复，因此更加智能化；

[0033] (3) 本发明方法具有更小体积，传统的故障恢复手段采用外部专用配置芯片，而本发明方法实现的故障检测和故障恢复都在 SOPC 芯片内部实现，有效地减少了系统的体积和功耗；

[0034] (4) 本发明方法可靠性更高。传统的故障恢复手段采用外部专用配置芯片，配置芯片和 SoPC 芯片的连接是在板级上进行实现，数据通路很容易暴露在空间辐射下，本发明方法在芯片内部实现故障检测和恢复，有利于抗辐加固设计的实现，同时防止了配置芯片在飞行过程中因振动造成的焊点脱落的问题，可靠性更高。

附图说明

[0035] 图 1 为本发明方法故障定位操作示意图；

[0036] 图 2 为本发明方法自重构操作示意图；

[0037] 图 3 为本发明方法 SoPC 芯片自重构操作示意图。

具体实施方式

[0038] 本发明对国产 SoPC 芯片 BM3109 的自主化容错设计进行研究，寻求在不增加 SoPC 芯片面积的前提下，基于“回读 - 重构”的方式，实现芯片的自主故障检测及故障修复。通过突破此项关键技术，将有利提高 SoPC 芯片在外太空环境应用中的可靠性，有利于推动国产 SoPC 芯片的发展。本发明为国产 SoPC 芯片 BM3109 提供一种自主容错和故障恢复的能力，在不增加外围检测设备和检测电路的情况下，基于“回读 - 自重构”的容错机制，使 BM3109 能够在空间环境下，自主完成故障检测、故障判读、故障修复。

[0039] 故障检测：启动配置 BM3109 中 Flash 的控制引脚，在时钟 CCLK 的控制下，同时读

取 BM3109 中 FPGA 配置存储器中的配置数据和 Flash 中存储的原始配置数据，并将两者逐位进行比较。当发现两者数据不匹配时，向 V8 处理器发送故障信号；

[0040] 故障判读：配置数据的验证是通过将回读的配置数据与下载到 FPGA 中的原始配置数据进行比较来实现的，基于比较文件格式上的差异能够验证回读的配置数据是否发生了故障；

[0041] 故障修复：FPGA 的功能故障主要是因 FPGA 中配置信息的反转造成，需要根据原始配置文件将其纠正。

[0042] 下面对基于回读自重构的 SoPC 芯片自容错方法详细说明。

[0043] 1 硬件连接

[0044] (1) 模式配置接口硬件连接

[0045] 将 FPGA 的模式选择引脚 M0、M1、M2 均连接到 GND 端；

[0046] (2) 回读接口硬件连接

[0047] 将 SPARC V8 处理器的 GPIO-PIO48 引脚连接到 FPGA 的 IO_D7 引脚，GPIO-PIO49 引脚连接到 FPGA 的 IO_D6 引脚，GPIO-PIO50 引脚连接到 FPGA 的 IO_D5 引脚，GPIO-PIO51 引脚连接到 FPGA 的 IO_D4 引脚，GPIO-PIO52 引脚连接到 FPGA 的 IO_D3 引脚，GPIO-PIO53 引脚连接到 FPGA 的 IO_D2 引脚，GPIO-PIO54 引脚连接到 FPGA 的 IO_D1 引脚，GPIO-PIO55 引脚连接到 FPGA 的 IO_D0 引脚，GPIO-PIO62 连接到 FPGA 的 IO_WRITE 引脚，GPIO-PIO63 引脚连接到 FPGA 的 IO_CS 引脚，GPIO-PIO61 引脚连接到 FPGA 的 GCLK 引脚；如表所示

[0048]

信号种类	信号	SPARC V8 处理器引脚	FPGA 引脚
时钟信号	GCLK	GPIO-PIO61	GCLK
读写控制信号	WRITE	GPIO-PIO62	IO_WRITE
片选控制信号	CS	GPIO-PIO63	IO_CS
8 路数据信号		GPIO-PIO48	IO_D7
		GPIO-PIO49	IO_D6
		GPIO-PIO50	IO_D5
		GPIO-PIO51	IO_D4
		GPIO-PIO52	IO_D3
		GPIO-PIO53	IO_D2
		GPIO-PIO54	IO_D1
		GPIO-PIO55	IO_D0

[0049] (3)FLASH 接口硬件连接

[0050] 将 SPARC V8 处理器地址线 Ai 接到 FLASH 的地址线 PAi-1，i = (1,2,3,,,22)，数据线 Dj，j = (16,17,18,,,31)，连接到 FLASH 的数据线 DQg，g = (0,1,2,,,15)，写控制信号端连接到 FLASH 的 WE 端，复位控制信号端连接到 FLASH 的 PRESET 端，片选控制信号端连接到 FLASH 的 CE 端，读控制信号端连接到 FLASH 的 OE 端；

[0051] (4)SDRAM 接口硬件连接

[0052] 将 SPARC V8 处理器地址线 Ai 接到 SDRAM 的地址线 PAi-1，i = (1,2,3,,,22)，数据线 Dj，j = (16,17,18,,,31)，连接到 SDRAM 的数据线 DQk，k = (0,1,2,,,31)，写控制信号端连接到 SDRAM 的 SDWEN 端，复位控制信号端连接到 FLASH 的 SDRASN 端，片选控制信号端

连接到 SDRAM 的 SDCS0 端，字节控制信号 BE0、BE1、BE2、BE3，依次连接到 SDDQM0、SDDQM1、SDDQM2、SDDQM3 ；

[0053] (5) 自重构接口硬件连接

[0054] 将 SPARC V8 处理器的 GPIO-PIO56 引脚连接到 FPGA 的 INIT 引脚，GPIO-PIO58 引脚连接到 FPGA 的 DONE 引脚，GPIO-PIO60 引脚连接到 FPGA 的 IO_DOUT_BUSY 引脚，GPIO-PIO57 引脚连接到 FPGA 的 PROGRAM 引脚 ；

[0055] 2 回读功能实现

[0056] (21) 根据 Xilinx 手册，生成回读命令数组。如表所示

[0057]

数据类型	数据值
同步字	0xAA99 5566h
写入到 FAR 寄存器	0x3000 2001h
起始帧地址	0x0000 0000h
写入到 CMD 寄存器	0x3000 8001h
包数据 RCFG	0x0000 0000h
从 FDRO 寄存器中读出	0x2800 6000h
数据字	0x4800 cb07h

[0058] (22) 配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，使 FPGA 的 IO_CS 输入为低电平 ；

[0059] (23) 经过 1 个时钟周期后，配置 SPARC V8 处理器 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，使 FPGA 的 IO_WRITE 引脚为低电平 ；

[0060] (24) 遍历回读命令数组并根据回读命令数组，连续配置 SPARC V8 处理器中的 GPIO 寄存器使 GPIO 组中的 15 个引脚生成时钟信号、读写控制信号、片选控制信号、配置使能信号、八路数据信号，送至 FPGA。

[0061] a. 通过周期性配置 SPARC V8 处理器 GPIO-PIO61 引脚寄存器，通过控制寄存器和数据寄存器的变化来为 FPGA 的时钟信号 GCLK 提供输入。每间隔 1 个时钟周期，交替的令 SPARC V8 处理器的 GPIO-PIO61 管脚输出“1”和“0”。每一个交替周期内，由三部分操作构成 ：配置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出 ；向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，使 FPGA 的 GCLK 输入为高电平 ；下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，使 FPGA 的 GCLK 输入为低电平。

[0062] b. 配置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，使 FPGA 的 IO_CS 输入为低电平 ；配置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，使 FPGA 的 IO_WRITE 引脚为低电平 ；配置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC

V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”，使 FPGA 的 PROGRAM 引脚为高电平。

[0063] c. 从回读命令数组中依次取出数据，在 CCLK 信号为高电平时，以 2 进制的格式，从高到低写入 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55，使 FPGA 的 8 路 IO_D7、IO_D6、IO_D5、IO_D4、IO_D3、IO_D2、IO_D1、IO_D0 信号接收到回读命令。重复执行此过程，直至回读命令数组被全部遍历。

[0064] (25) 配置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，使 FPGA 的 IO_WRITE 引脚输入为高电平；配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“1”，使 FPGA 的 IO_CS 引脚输入为高电平；

[0065] (26) 配置 SPARC V8 处理器 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 引脚为输入，以读取 FPGA 数据。

[0066] (27) 配置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，使 FPGA 的 IO_WRITE 引脚输入为高电平；配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“1”，使 FPGA 的 IO_CS 引脚输入为高电平；配置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出；向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，使 FPGA 的 GCLK 输入为低电平；下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，使 FPGA 的 GCLK 输入为高电平。

[0067] (28) 配置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，使 FPGA 的 IO_WRITE 引脚输入为高电平；配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，使 FPGA 的 IO_CS 引脚输入为低电平；

[0068] (29) 根据回读长度，连续配置 SPARC V8 处理器中的 GPIO 寄存器使 GPIO 组中的 15 个引脚生成时钟信号、读写控制信号、片选控制信号、配置使能信号、八路数据信号，送至 FPGA。

[0069] a. 通过周期性配置 SPARC V8 处理器 GPIO-PIO61 引脚寄存器，通过控制寄存器和数据寄存器的变化来为 FPGA 的时钟信号 GCLK 提供输入。每间隔 1 个时钟周期，交替的令 SPARC V8 处理器的 GPIO-PIO61 管脚输出“1”和“0”。每一个交替周期内，由三部分操作构成：配置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出；向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，使 FPGA 的 GCLK 输入为高电平；下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，使 FPGA 的 GCLK 输入为低电平。

[0070] b. 配置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“0”，使 FPGA 的 IO_CS 输入为低电平；配置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，使 FPGA 的 IO_WRITE 引脚为高电平；配置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”，使 FPGA 的 PROGRAM 引脚为高电平。

[0071] c. 从 FPGA 依次取出数据，在 CCLK 信号为高电平时，以 2 进制的格式，从 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、

GPIO-PIO54、GPIO-PIO55 读取数据，从而获取 FPGA 的 8 路 IO_D7、IO_D6、IO_D5、IO_D4、IO_D3、IO_D2、IO_D1、IO_D0 数据。重复执行此过程，直至 FPGA 数据全部读出。

[0072] (210) 配置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，使 FPGA 的 IO_WRITE 引脚输入为高电平；配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“1”，使 FPGA 的 IO_CS 引脚输入为高电平；回读完成。

[0073] 通过数据接口回读得到的总字节数共计 207900，由 2747 个 CLB 帧构成，其中每帧有 80 个字节，共有 197920 个字节，其中，填充字节数为 9980 个。

[0074] (211) 通过故障检测机制监测到故障后，需要对故障进行定位和分析。由于芯片在空间轨道上运行，无法使用现有的故障检测设备施加测试向量，芯片内部亦不支持自检测电路。内部配置存储器被分成若干块，每一块的配置信息以“帧”为最小配置单位。实际写入配置存储器的位流的部分被称作“数据帧”。将回读得到的数据帧数组与存储在外部存储器中的原始有效配置数据进行比较，当数据出现不一致时，说明配置出现错误，而不一致的数据所在的数据帧即为故障帧。确定了具体故障帧后，可根据故障帧的帧标识，通过确定的行、列地址计算公式计算得到包括故障帧行、列坐标信息，通过对故障帧中错误数据位置的解读，亦可判断出现故障的区域特征，如图 1 所示。

[0075] (212) 由于 FPGA 中配置信息的反转导致的故障需要根据原始配置文件将其纠正。通过 V8 处理器中，使用 GPIO 接口，采用 SoPC 芯片自主重构软配置方法，从 FLASH 中读取原始配置文件，并通过软配置接口，对 FPGA 进行重新配置，执行一次全局重构时间仅为 40ms 左右。由此，即可在不增加外置电路和其他一些设备的前提下，纠正 SoPC 芯片内部的软故障。

[0076] SoPC 芯片自主重构软配置方法通过在 SoPC 芯片 BM3109 内部，构建配置数据链路，实现 SoPC 芯片的自主重构。SoPC 芯片 BM3109 根据外界环境的变化和任务的需求，从芯片内部的存储空间内读取 FPGA 的配置文件，经软配置接口将配置数据加载到 FPGA 中，自重构操作如图 2 所示，SoPC 芯片自重构操作如图 3 所示。其中，BM3109 包括外部存储器 FLASH、SPARC V8 处理器和 FPGA。

[0077] 3、软配置过程

[0078] (31) 将 ISE10.1 环境任意生成的 .bit 文件到外部存储器 FLASH 中。对 SPARC V8 处理器、FLASH、FPGA 芯片进行上电，上电完成后，SPARC V8 处理器从 FLASH 中读取 .bit 文件，参照 Xilinx 公司 .bit 文件格式，去掉用文件的头部无效信息，提取有效数据信息，并按照 Xilinx Virtex 芯片的配置格式，生成配置数组。

[0079] (32) 配置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，先向 GPIO-PIO57 引脚的数据寄存器中写入“1”，使 FPGA 芯片的 PROGRAM 引脚输入为高电平。一个时钟周期后，向 SPARC V8 处理器的 GPIO-PIO57 引脚的数据寄存器中写入“0”，使 FPGA 的 PROGRAM 引脚输入为低电平来启动复位配置逻辑。

[0080] (33)2us 后，配置 SPARC V8 处理器的 GPIO-PIO56 为输入，使其持续监测 FPGA 的 INIT 引脚的电压变化，当 INIT 引脚变由低电平变为高电平时，即 GPIO-PIO56 的输入由“0”变为“1”时，表示清空 FPGA 内部寄存器操作完成；如果 FPGA 的 INIT 引脚为高电平，则重复步骤 (31)- 步骤 (32)，直至 FPGA 的 INIT 引脚出现由低电平向高电平的跳变；

[0081] (34) 配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，使 FPGA 的 IO_CS 输入为低电平；

[0082] (35) 经过 1 个时钟周期后，配置 SPARC V8 处理器 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO 的数据寄存器写入“0”，使 FPGA 的 IO_WRITE 引脚为低电平；

[0083] (36) 遍历配置信息数组并根据配置信息数组，连续配置 SPARC V8 处理器中的 GPIO 寄存器使 GPIO 组中的 15 个引脚生成时钟信号、读写控制信号、片选控制信号、配置使能信号、八路数据信号，送至 FPGA，其中 15 路信号如表所示。

[0084]

信号种类	信号	SPARC V8 处理器引脚	FPGA 引脚
时钟信号	GCLK	GPIO-PIO61	GCLK
读写控制信号	WRITE	GPIO-PIO62	IO_WRITE
片选控制信号	CS	GPIO-PIO63	IO_CS
配置使能信号	PROGRAM	GPIO-PIO57	PROGRAM
	BUSY	GPIO-PIO60	IO_DOUT_BUSY
	INIT	GPIO-PIO56	INIT
	DONE	GPIO-PIO58	DONE
8 路数据信号		GPIO-PIO48	IO_D7
		GPIO-PIO49	IO_D6
		GPIO-PIO50	IO_D5
		GPIO-PIO51	IO_D4
		GPIO-PIO52	IO_D3
		GPIO-PIO53	IO_D2
		GPIO-PIO54	IO_D1
		GPIO-PIO55	IO_D0

[0085] a. 通过周期性配置 SPARC V8 处理器 GPIO-PIO61 引脚寄存器，通过控制寄存器和数据寄存器的变化来为 FPGA 的时钟信号 GCLK 提供输入。每间隔 1 个时钟周期，交替的令 SPARC V8 处理器的 GPIO-PIO61 管脚输出“1”和“0”。每一个交替周期内，由三部分操作构成：配置 SPARC V8 处理器的 GPIO-PIO61 引脚为输出；向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“1”，使 FPGA 的 GCLK 输入为高电平；下一个时钟周期向 SPARC V8 处理器的 GPIO-PIO61 的数据寄存器写入“0”，使 FPGA 的 GCLK 输入为低电平。

[0086] b. 配置 SPARC V8 处理器的 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“0”，使 FPGA 的 IO_CS 输入为低电平；配置 SPARC V8 处理器的 GPIO-PIO62 脚为输出，并向 SPARC V8 处理器 GPIO 的数据寄存器写入“0”，使 FPGA 的 IO_WRITE 引脚为低电平；配置 SPARC V8 处理器的 GPIO-PIO57 引脚为输出，并向 SPARC V8 处理器的 GPIO-PIO57 的数据寄存器写入“1”，使 FPGA 的 PROGRAM 引脚为高电平。

[0087] c. 从配置信息数组中依次取出数据，在 CCLK 信号为高电平时，以 2 进制的格式，从高到低写入 SPARC V8 处理器的 GPIO-PIO48、GPIO-PIO49、GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55，使 FPGA 的 8 路 IO_D7、IO_D6、IO_D5、IO_D4、IO_D3、IO_D2、IO_D1、IO_D0 信号接收到配置数据。重复执行此过程，直至配置信息数组被全部遍历。

[0088] d. 执行 (c) 过程中，持续监控 FPGA 的 IO_DOUT_BUSY 引脚，若 IO_DOUT_BUSY 为高电平时，此时的配置数据将不能够被 FPGA 识别，需要持续向 GPIO-PIO48、GPIO-PIO49、

GPIO-PIO50、GPIO-PIO51、GPIO-PIO52、GPIO-PIO53、GPIO-PIO54、GPIO-PIO55 的数据寄存器中写入当前配置数据，直至 IO_DOUT_BUSY 信号输出为低电平。

[0089] (37) 配置 SPARC V8 处理器 GPIO-PIO62 引脚为输出，向 SPARC V8 处理器 GPIO-PIO62 的数据寄存器写入“1”，使 FPGA 的 IO_WRITE 引脚输入为高电平；配置 SPARC V8 处理器 GPIO-PIO63 引脚为输出，向 SPARC V8 处理器 GPIO-PIO63 的数据寄存器写入“1”，使 FPGA 的 IO_CS 引脚输入为高电平；

[0090] (38) 配置 SPARC V8 处理器 GPIO-PIO58 引脚为输入，持续检测 FPGA 的 DONE 控制信号的输出，如果电平为高说明配置完成，否则继续等待，直至 DONE 信号为高。若等待时间超出 1s 的阈值，则重复执行 (31) 至 (37)。

[0091] 本发明说明书中未作详细描述的内容属本领域技术人员的公知技术。

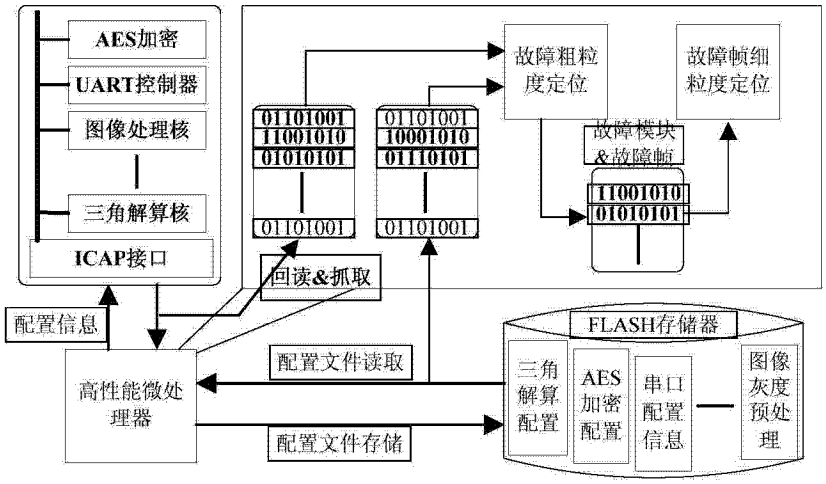

AES加密
UART控制器
图像处理核
三角解算核
ICAP接口
回读&抓取
故障粗粒度定位
故障帧细粒度定位
故障模块&故障帧
配置信息
高性能微处理器
配置文件读取
配置文件存储
FLASH存储器
三角解算配置
AES加密配置
串口配置信息
图像灰度预处理

图 1

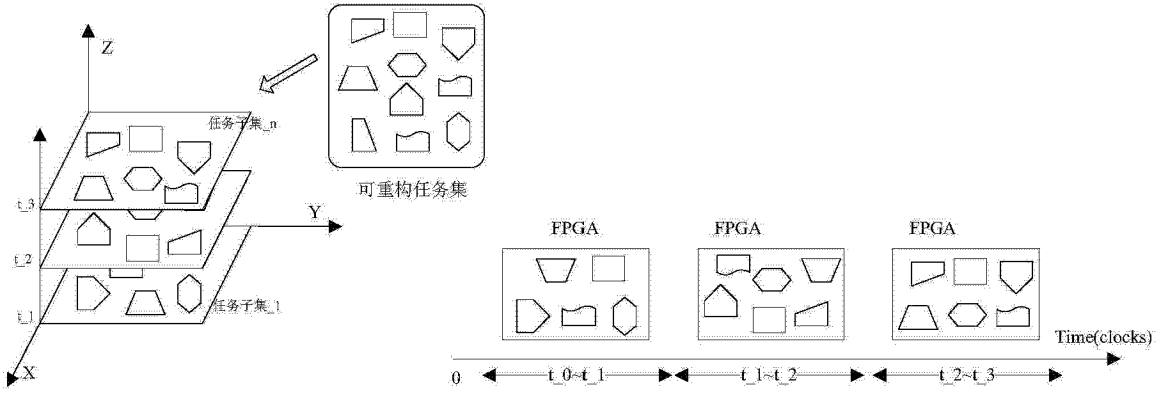

Z
Y
X
任务子集_n
任务子集_1
可重构任务集
FPGA
FPGA
FPGA
Time(clocks)
0
t_0~t_1
t_1~t_2
t_2~t_3

图 2

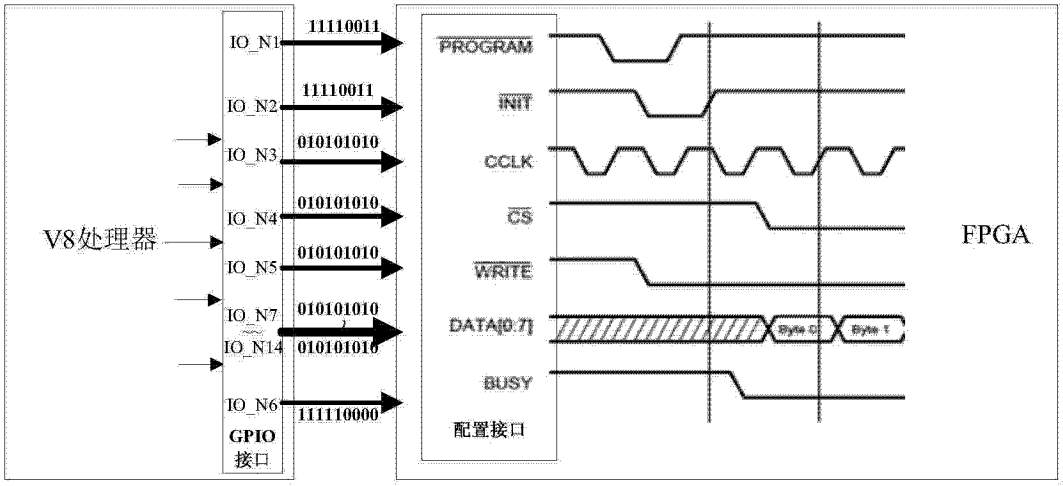

V8处理器
IO_N1
IO_N2
IO_N3
IO_N4
IO_N5
IO_N7
IO_N14
IO_N6
GPIO接口
11110011
11110011
010101010
010101010
010101010
010101010
010101010
111110000
PROGRAM
INIT
CCLK
CS
WRITE
DATA[0:7]
BUSY
配置接口
FPGA

图 3