(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 116450552 B
(45)授权公告日 2023.08.29

(21)申请号 202310679072.2

(22)申请日 2023.06.09

(65)同一申请的已公布的文献号
申请公布号 CN 116450552 A

(43)申请公布日 2023.07.18

(73)专利权人 江苏润石科技有限公司
地址 214000 江苏省无锡市新吴区弘毅路10号金乾座1901-1910、2001-2010室
专利权人 上海矽朔微电子有限公司

(72)发明人 宋顺涛 张明

(74)专利代理机构 南京经纬专利商标代理有限公司 32200
专利代理师 阚梦诗

(51)Int.Cl.
G06F 13/20(2006.01)
G06F 13/42(2006.01)
G06F 15/17(2006.01)

(56)对比文件
CN 104346131 A,2015.02.11
CN 103714036 A,2014.04.09
CN 108228520 A,2018.06.29
CN 1558332 A,2004.12.29

审查员 黄旭光

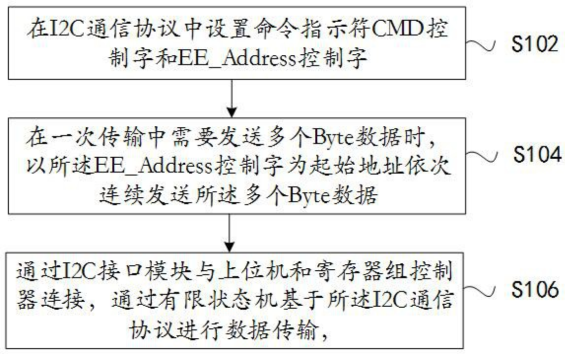
权利要求书3页 说明书8页 附图3页

(54)发明名称

基于I2C总线异步批量读写寄存器的方法及系统

(57)摘要

本申请实施例提供了一种基于I2C总线异步批量读写寄存器的方法及系统,通过在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字,其中,所述CMD控制字配置为指示Byte的标志位,所述EE_Address控制字配置为指示要访问的寄存器的地址,且配置为指示在一次传输中发送Byte数据的起始地址;在一次传输中需要发送多个Byte数据时,以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据,解决了相关技术中的I2C单Byte的传输方式传输效率低下的问题。



1.一种基于I2C总线异步批量读写寄存器的方法,其特征在于,包括:

在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字,其中,所述CMD控制字配置为指示Byte的标志位,所述EE_Address控制字配置为指示要访问的寄存器的地址,且配置为指示在一次传输中发送Byte数据的起始地址;

在一次传输中需要发送多个Byte数据时,以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据;

通过I2C接口模块与上位机和寄存器组控制器连接,通过有限状态机基于所述I2C通信协议进行数据传输,其中,所述I2C接口模块设置在有限状态机上,所述有限状态机设置在从机侧,所述I2C接口模块包括SCL接口、SDI接口、SDO接口、RST接口和控制器总线接口,所述寄存器组控制器配置为控制一组寄存器,所述SCL接口为串行时钟接口,所述SDI接口为数据输入接口,所述SDO接口为数据输出接口,所述RST接口为复位接口;

所述有限状态机基于所述I2C通信协议进行数据传输包括以下步骤:

S1,所述有限状态机在空闲状态时为IDLE状态,当检测到开始信号时,进入SLAVE_ADDR状态,此时接收从机的地址;

S2,在接收8bit数据之后,所述有限状态机进入SLAVE_ACK状态,若所述上位机发送的地址与从机吻合,则应答ACK,之后根据所述I2C通信协议,进入WRITE_CMD状态,此状态用于接收当前操作为单Byte操作还是多Byte操作,以及接收当前操作的初始地址;

S3,在所述WRITE_CMD状态结束后,所述有限状态机进入SLAVE_ACK状态进行应答;

S4,若为单Byte操作,则首先进入WRITE_Byte操作,即为写单Byte状态,若在此时检测到开始信号,则进入READ_SLAVE_ADDR状态,继续接收一次从机地址,之后进入READ_Byte状态,读单Byte数据,之后进入MASTER_ACK,即等待上位机接收读数据之后给出应答以及停止信号,传输结束;

S5,若未检测到开始信号,则进行WRITE_Byte,写完数据之后进行SLAVE_ACK,收到停止信号后传输结束;

S6,若为多Byte操作,则首先进入WRITE_CNT状态,若在此时检测到了开始信号,则进入READ_CNT状态,在读出CNT之后,继续读CNT个Byte结束传输,每次读完Byte之后进入MASTER_ACK,若当前完成Byte数小于READ_CNT阶段传输的CNT,则继续回到READ_BLOCK,若传输完CNT个Byte,回到IDLE;

S7,若未检测到开始信号,首先写WRITE_CNT,之后再写CNT个Byte之后结束传输,每次写完Byte之后进入SLAVE_ACK,若当前完成Byte数小于WRITE_CNT阶段传输的CNT,则继续回到WRITE_BLOCK,若传输完CNT个Byte,回到IDLE。

2. 根据权利要求1所述的方法,其特征在于,所述方法还包括:

基于所述I2C通信协议执行写操作时,在串行时钟SCL信号的上升沿获取数据,基于所述I2C通信协议执行读操作时,在所述SCL信号的下降沿发送数据;或

基于所述I2C通信协议执行写操作时,在串行时钟SCL信号的下降沿获取数据,基于所述I2C通信协议执行读操作时,在所述SCL信号的上升沿发送数据。

3.根据权利要求1所述的方法,其特征在于,所述方法还包括:

所述上位机通过所述SCL接口向所述有限状态机发送时钟信号;

所述上位机通过所述SDI接口和所述SDO接口与所述有限状态机进行双向数据通信;以

及

所述上位机通过所述RST接口向所述有限状态机发送复位信号。

4.根据权利要求1所述的方法,其特征在于,所述方法还包括:

在所述控制器总线接口和所述寄存器组控制器之间设置通信总线,其中,所述通信总线配置为传输以下至少之一的数据信号:

第一数据信号,配置为指示所述上位机要访问的寄存器的地址;

第二数据信号,配置为指示一次传输的数据的Byte数;

第三数据信号,配置为指示从I2C总线上接收到的8bit数据;

第四数据信号,配置为指示将要通过I2C总线发送出去的8bit数据;

第五数据信号,配置为当需要写数据时,产生一个所述第五数据信号的脉冲;

第六数据信号,配置为当需要发送数据时,将所述第六数据信号拉高一个脉冲。

5.一种基于I2C总线异步批量读写寄存器的系统,其特征在于,所述系统包括:有限状态机和寄存器组控制器,I2C接口模块设置在有限状态机上,其中,在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字,所述CMD控制字配置为指示Byte的标志位,所述EE_Address控制字配置为指示要访问的寄存器的地址,且配置为指示在一次传输中发送Byte数据的起始地址;在一次传输中需要发送多个Byte数据时,以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据;

所述有限状态机基于所述I2C通信协议进行数据传输包括以下步骤:

S1,所述有限状态机在空闲状态时为IDLE状态,当检测到开始信号时,进入SLAVE_ADDR状态,此时接收从机的地址;

S2,在接收8bit数据之后,所述有限状态机进入SLAVE_ACK状态,若上位机发送的地址与从机吻合,则应答ACK,之后根据所述I2C通信协议,进入WRITE_CMD状态,此状态用于接收当前操作为单Byte操作还是多Byte操作,以及接收当前操作的初始地址;

S3,在所述WRITE_CMD状态结束后,所述有限状态机进入SLAVE_ACK状态进行应答;

S4,若为单Byte操作,则首先进入WRITE_Byte操作,即为写单Byte状态,若在此时检测到开始信号,则进入READ_SLAVE_ADDR状态,继续接收一次从机地址,之后进入READ_Byte状态,读单Byte数据,之后进入MASTER_ACK,即等待上位机接收读数据之后给出应答以及停止信号,传输结束;

S5,若未检测到开始信号,则进行WRITE_Byte,写完数据之后进行SLAVE_ACK,收到停止信号后传输结束;

S6,若为多Byte操作,则首先进入WRITE_CNT状态,若在此时检测到了开始信号,则进入READ_CNT状态,在读出CNT之后,继续读CNT个Byte结束传输,每次读完Byte之后进入MASTER_ACK,若当前完成Byte数小于READ_CNT阶段传输的CNT,则继续回到READ_BLOCK,若传输完CNT个Byte,回到IDLE;

S7,若未检测到开始信号,首先写WRITE_CNT,之后再写CNT个Byte之后结束传输,每次写完Byte之后进入SLAVE_ACK,若当前完成Byte数小于WRITE_CNT阶段传输的CNT,则继续回到WRITE_BLOCK,若传输完CNT个Byte,回到IDLE。

6.根据权利要求5所述的系统,其特征在于,

所述I2C接口模块与上位机和寄存器组控制器连接,所述I2C接口模块包括SCL接口、

SDI接口、SDO接口、RST接口和控制器总线接口，所述寄存器组控制器配置为控制一组寄存器，所述SCL接口为串行时钟接口，所述SDI接口为数据输入接口，所述SDO接口为数据输出接口，所述RST接口为复位接口。

7.根据权利要求6所述的系统，其特征在于，所述系统配置为：

总线状态检测模块接收SCL信号，并输出信号至有限状态机，有限状态机输出信号至输入输出模块，输入输出模块输出信号至总线状态检测模块，并与SDA信号通信连接，SDA信号包括SDI信号和SDO信号，其中，SDA信号为数据传输信号，SDI信号为数据输入信号，SDO信号为数据输出信号；

寄存器组控制器内部包括读写信号检测模块、片内地址选择模块、I2C寄存器组、EEPROM控制器、EE寄存器组和输出多路选择器，其中，读写信号检测模块输出信号至片内地址选择模块，片内地址选择模块输出信号至I2C寄存器组，I2C寄存器组输出信号至输出多路选择器，EEPROM控制器输出信号至EE寄存器组，EE寄存器组输出信号至输出多路选择器，输出多路选择器输出模拟控制信号至模拟电路单元，EEPROM控制器通过EEPROM接口输出控制信号至EEPROM。

8.根据权利要求7所述的系统，其特征在于，所述系统包含一个EEPROM，在存在主时钟时能够通过主时钟控制进行读取EEPROM的值将其存放在EE寄存器组中，能够通过I2C协议修改的寄存器组为I2C寄存器组，所述EE寄存器组和所述I2C寄存器组大小相同；

当存在主时钟，无I2C时钟时，通过主时钟将用户事先烧写配置好的EEPROM中的值读出，存放在EE寄存器组之中，同时多路选择器选择EE寄存器组的值输出给模拟电路单元；

当存在I2C时钟时，无论主时钟是否存在，首先将EE寄存器组的值赋值给I2C寄存器组，I2C协议在EE寄存器组的值的基础上对模拟进行配置，同时输出多路选择器选择I2C寄存器组的值输出给模拟电路单元。

基于I2C总线异步批量读写寄存器的方法及系统

技术领域

[0001] 本申请涉及微电子技术领域，具体而言，涉及一种基于I2C总线异步批量读写寄存器的方法及系统。

背景技术

[0002] 目前市场对于芯片的功能和性能提出越来越高的要求，因此为了更好地控制芯片的功能，尤其是芯片内部模拟部分的工作模式以及相应的修调值，提高其灵活性。往往需要很多相应的寄存器来对芯片功能进行相关的配置，为了保持良好的兼容性，这些寄存器一般通过PHILIPS I2C接口来配置。然而上述现有技术存在如下技术问题：芯片配置大多采用同步设计，非常依赖系统主时钟，并且随之带来的问题是功耗的增加，导致对于功耗面积要求很高的移动设备以及视频监控设备不太理想。同时依赖主时钟会导致当芯片晶振出现问题时，对芯片的其他模拟模块失去了配置以及debug的能力。除此之外，当前I2C单Byte的传输方式传输效率低，当存在多个寄存器需要配置时，这种传输方式效率较为低下。

[0003] 针对相关技术中，I2C单Byte的传输方式传输效率低的问题，目前尚未有有效的解决办法。

发明内容

[0004] 本申请实施例提供了一种基于I2C总线异步批量读写寄存器的方法及系统，以至少解决相关技术中的I2C单Byte的传输方式传输效率低的问题。

[0005] 在本申请的一个实施例中，提出了一种基于I2C总线异步批量读写寄存器的方法，包括：在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字，其中，所述CMD控制字配置为指示Byte的标志位，所述EE_Address控制字配置为指示要访问的寄存器的地址，且配置为指示在一次传输中发送Byte数据的起始地址；在一次传输中需要发送多个Byte数据时，以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据；通过I2C接口模块与上位机和寄存器组控制器连接，通过有限状态机基于所述I2C通信协议进行数据传输，其中，所述I2C接口模块设置在有限状态机上，所述有限状态机设置在从机侧，所述I2C接口模块包括SCL接口、SDI接口、SDO接口、RST接口和控制器总线接口，所述寄存器组控制器配置为控制一组寄存器；所述有限状态机基于所述I2C通信协议进行数据传输包括以下步骤：S1，所述有限状态机在空闲状态时为IDLE状态，当检测到开始信号时，进入SALVE_ADDR状态，此时接收从机的地址；S2，在接收8bit数据之后，所述有限状态机进入SLAVE_ACK状态，若所述上位机发送的地址与从机吻合，则应答ACK，之后根据所述I2C通信协议，进入WRITE_CMD状态，此状态用于接收当前操作为单Byte操作还是多Byte操作，以及接收当前操作的初始地址；S3，在所述WRITE_CMD状态结束后，所述有限状态机进入SLAVE_ACK状态进行应答；S4，若为单Byte操作，则首先进入WRITE_Byte操作，即为写单Byte状态，若在此时检测到开始信号，则进入READ_SLAVE_ADDR状态，继续接收一次从机地址，之后进入READ_Byte状态，读单Byte数据，之后进入MASTER_ACK，即等待上位机接收读数据之后给

出应答以及停止信号，传输结束；S5，若未检测到开始信号，则进行WRITE_Byte，写完数据之后进行SALVE_ACK，收到停止信号后传输结束；S6，若为多Byte操作，则首先进入WRITE_CNT状态，若在此时检测到了开始信号，则进入READ_CNT状态，在读出CNT之后，继续读CNT个Byte结束传输；S7，若未检测到开始信号，首先写WRITE_CNT，之后再写CNT个Byte之后结束传输。在一实施例中，所述方法还包括：基于所述I2C通信协议执行写操作时，在串行时钟SCL信号的上升沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的下降沿发送数据；或基于所述I2C通信协议执行写操作时，在串行时钟SCL信号的下降沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的上升沿发送数据。

[0006] 在一实施例中，所述方法还包括：所述上位机通过所述SCL接口向所述有限状态机发送时钟信号；所述上位机通过所述SDI接口和所述SDO接口与所述有限状态机进行双向数据通信；以及所述上位机通过所述RST接口向所述有限状态机发送复位信号。

[0007] 在一实施例中，所述方法还包括：在所述控制器总线接口和所述寄存器组控制器之间设置通信总线，其中，所述通信总线配置为传输以下至少之一的数据信号：第一数据信号，配置为指示所述上位机要访问的寄存器的地址；第二数据信号，配置为指示一次传输的数据的Byte数；第三数据信号，配置为指示从I2C总线上接收到的8bit数据；第四数据信号，配置为指示将要通过I2C总线发送出去的8bit数据；第五数据信号，配置为当需要写数据时，产生一个所述第五数据信号的脉冲；第六数据信号，配置为当需要发送数据时，将所述第六数据信号拉高一个脉冲。

[0008] 在本申请的一个实施例中，还提出了一种基于I2C总线异步批量读写寄存器的系统，其特征在于，所述系统包括：有限状态机和寄存器组控制器，I2C接口模块设置在有限状态机上，其中，在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字，所述CMD控制字配置为指示Byte的标志位，所述EE_Address控制字配置为指示要访问的寄存器的地址，且配置为指示在一次传输中发送Byte数据的起始地址；在一次传输中需要发送多个Byte数据时，以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据； 所述有限状态机基于所述I2C通信协议进行数据传输包括以下步骤：S1，所述有限状态机在空闲状态时为IDLE状态，当检测到开始信号时，进入SALVE_ADDR状态，此时接收从机的地址；S2，在接收8bit数据之后，所述有限状态机进入SLAVE_ACK状态，若所述上位机发送的地址与从机吻合，则应答ACK，之后根据所述I2C通信协议，进入WRITE_CMD状态，此状态用于接收当前操作为单Byte操作还是多Byte操作，以及接收当前操作的初始地址；S3，在所述WRITE_CMD状态结束后，所述有限状态机进入SLAVE_ACK状态进行应答；S4，若为单Byte操作，则首先进入WRITE_Byte操作，即为写单Byte状态，若在此时检测到开始信号，则进入READ_SLAVE_ADDR状态，继续接收一次从机地址，之后进入READ_Byte状态，读单Byte数据，之后进入MASTER_ACK，即等待上位机接收读数据之后给出应答以及停止信号，传输结束；S5，若未检测到开始信号，则进行WRITE_Byte，写完数据之后进行SALVE_ACK，收到停止信号后传输结束；S6，若为多Byte操作，则首先进入WRITE_CNT状态，若在此时检测到了开始信号，则进入READ_CNT状态，在读出CNT之后，继续读CNT个Byte结束传输；S7，若未检测到开始信号，首先写WRITE_CNT，之后再写CNT个Byte之后结束传输byte。在一实施例中，所述方法还包括：基于所述I2C通信协议执行写操作时，在串行时钟SCL信号的上升沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的下降沿发送数据；或基于所述I2C通信协议执行写

操作时，在串行时钟SCL信号的下降沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的上升沿发送数据。

[0009] 在一实施例中，所述I2C接口模块与上位机和寄存器组控制器连接，所述I2C接口模块包括SCL接口、SDI接口、SDO接口、RST接口和控制器总线接口，所述寄存器组控制器配置为控制一组寄存器，所述SCL接口为串行时钟接口，所述SDI接口为数据输入接口，所述SDO接口为数据输出接口，所述RST接口为复位接口。

[0010] 在一实施例中，所述系统配置为：总线状态检测模块接收SCL信号，并输出信号至有限状态机，有限状态机输出信号至输入输出模块，输入输出模块输出信号至总线状态检测模块，并与SDA信号通信连接，SDA信号包括SDI信号和SDO信号，其中，SDA信号为数据传输信号，SDI信号为数据输入信号，SDO信号为数据输出信号；寄存器组控制器内部包括读写信号检测模块、片内地址选择模块、I2C寄存器组、EEPROM控制器、EE寄存器组和输出多路选择器，其中，读写信号检测模块输出信号至片内地址选择模块，片内地址选择模块输出信号至I2C寄存器组，I2C寄存器组输出信号至输出多路选择器，EEPROM控制器输出信号至EE寄存器组，EE寄存器组输出信号至输出多路选择器，输出多路选择器输出模拟控制信号至模拟电路单元，EEPROM控制器通过EEPROM接口输出控制信号至EEPROM。

[0011] 在一实施例中，所述系统包含一个电可擦编程只读存储器(Electrically Erasable Programmable Read-Only Memory，简称为EEPROM)，在存在主时钟时能够通过主时钟控制进行读取EEPROM的值将其存放在EE寄存器组中，能够通过I2C协议修改的寄存器组为I2C寄存器组，所述EE寄存器组和所述I2C寄存器组大小相同；当存在主时钟，无I2C时钟时，通过主时钟将用户事先烧写配置好的EEPROM中的值读出，存放在EE寄存器组之中，同时多路选择器选择EE寄存器组的值输出给模拟电路单元；当存在I2C时钟时，无论主时钟是否存在，首先将EE寄存器组的值赋值给I2C寄存器组，I2C协议在EE寄存器组的值的基础上对模拟进行配置，同时输出多路选择器选择I2C寄存器组的值输出给模拟电路单元。

[0012] 通过本申请实施例提供的基于I2C总线异步批量读写寄存器的方法及系统，通过在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字，其中，所述CMD控制字配置为指示Byte的标志位，所述EE_Address控制字配置为指示要访问的寄存器的地址，且配置为指示在一次传输中发送Byte数据的起始地址；在一次传输中需要发送多个Byte数据时，以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据，解决了相关技术中的I2C单Byte的传输方式传输效率低下的问题，能够实现基于I2C协议的批量数据传输。

附图说明

[0013] 此处所说明的附图用来提供对本申请的进一步理解，构成本申请的一部分，本申请的示意性实施例及其说明用于解释本申请，并不构成对本申请的不当限定。在附图中：

[0014] 图1是根据本申请实施例的一种可选的基于I2C总线异步批量读写寄存器的方法流程图；

[0015] 图2是根据本申请实施例的一种可选的基于I2C总线异步批量读写寄存器的系统的结构示意图；

[0016] 图3是根据本申请实施例的一种可选的有限状态机的状态转移示意图；

[0017] 图4是根据本申请实施例的一种可选的多时钟域同时配置模拟寄存器的原理图。

具体实施方式

[0018] 下文中将参考附图并结合实施例来详细说明本申请。需要说明的是，在不冲突的情况下，本申请中的实施例及实施例中的特征可以相互组合。

[0019] 需要说明的是，本申请的说明书和权利要求书及上述附图中的术语“第一”、“第二”等是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。

[0020] 图1是根据本申请实施例的一种可选的基于I2C总线异步批量读写寄存器的方法流程图，包括以下步骤：

[0021] 步骤S102，在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字，其中，所述CMD控制字配置为指示Byte的标志位，所述EE_Address控制字配置为指示要访问的寄存器的地址，且配置为指示在一次传输中发送Byte数据的起始地址；

[0022] 步骤S104，在一次传输中需要发送多个Byte数据时，以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据；

[0023] 步骤S106，通过I2C接口模块与上位机和寄存器组控制器连接，通过有限状态机基于所述I2C通信协议进行数据传输，其中，所述I2C接口模块设置在有限状态机上，所述有限状态机设置在从机侧，所述I2C接口模块包括SCL接口、SDI接口、SDO接口、RST接口和控制器总线接口，所述寄存器组控制器配置为控制一组寄存器；所述有限状态机基于所述I2C通信协议进行数据传输包括以下步骤：S1，所述有限状态机在空闲状态时为IDLE状态，当检测到开始信号时，进入SALVE_ADDR状态，此时接收从机的地址；S2，在接收8bit数据之后，所述有限状态机进入SLAVE_ACK状态，若所述上位机发送的地址与从机吻合，则应答ACK，之后根据所述I2C通信协议，进入WRITE_CMD状态，此状态用于接收当前操作为单Byte操作还是多Byte操作，以及接收当前操作的初始地址；S3，在所述WRITE_CMD状态结束后，所述有限状态机进入SLAVE_ACK状态进行应答；S4，若为单Byte操作，则首先进入WRITE_Byte操作，即为写单Byte状态，若在此时检测到开始信号，则进入READ_SLAVE_ADDR状态，继续接收一次从机地址，之后进入READ_Byte状态，读单Byte数据，之后进入MASTER_ACK，即等待上位机接收读数据之后给出应答以及停止信号，传输结束；S5，若未检测到开始信号，则进行WRITE_Byte，写完数据之后进行SALVE_ACK，收到停止信号后传输结束；S6，若为多Byte操作，则首先进入WRITE_CNT状态，若在此时检测到了开始信号，则进入READ_CNT状态，在读出CNT之后，继续读CNT个Byte结束传输；S7，若未检测到开始信号，首先写WRITE_CNT，之后再写CNT个Byte之后结束传输。在一实施例中，所述方法还包括：基于所述I2C通信协议执行写操作时，在串行时钟SCL信号的上升沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的下降沿发送数据；或基于所述I2C通信协议执行写操作时，在串行时钟SCL信号的下降沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的上升沿发送数据。

[0024] 需要说明的是，CMD控制字可以是0也可以是1，0代表这次传输就只传输一个Byte，这样就不需要额外传输Byte_CNT了，1表示这次传输需要传输多个Byte，可以用Byte_CNT来指示Byte个数。

[0025] 多个比特的数据从起始地址一次写入，比如一次传输四个Byte，以EE_Address控制字为起始地址，四个Byte依次写到EE_Address，EE_Address+1，EE_Address+2，EE_

Address+3。

[0026]　根据需求，可以选择单Byte读写操作以及多Byte读写操作，其中Byte操作的数据协议格式如下，其中CMD为Byte以及多Byte的标志位。

[0027]　单Byte操作的数据协议格式如下：

[0028]　单Byte读协议如下表1示：

[0029]　表1

[0030]

1	7	1	1	1	7	1	8	1	1
Start	Slave_ Address	W	ACK	CMD	EE_Address	ACK	Byte_Data	ACK	Stop

[0031]　单Byte写协议如下表2：

[0032]　表2

[0033]

1	7	1	1	1	7	1	1	7	1	8	1	1
Start	Slave Address	W	ACK	CMD	EE_Address	ACK	Repeat Start	Slave Address	R	Byte_Data	ACK	Stop

[0034]　多Byte操作的数据协议格式如下：

[0035]　多Byte写协议如下表3所示：

[0036]　表3

[0037]

1	7	1	1	1	7	1	7	1
Start	Slave Address	W	ACK	CMD	EE_Address	ACK	Byte CNT	ACK

8	1	8	1		8	1	1
Byte Data 0	ACK	Byte Data 1	ACK	……	Byte Data N	ACK	Stop

[0038]　多Byte读协议如下表4所示：

[0039]　表4

[0040]

1	7	1	1	1	7	1	1	7	1	1
Start	Slave Address	W	ACK	CMD	EE_Address	ACK	Repeat Start	Slave Address	R	ACK

7	1	8	1	8	1		8	1	1
Byte CNT	ACK	Byte Data 0	ACK	Byte Data 1	ACK	……	Byte Data N	ACK	Stop

[0041]　在一实施例中，所述方法还包括：基于所述I2C通信协议执行写操作时，在SCL信号的上升沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的下降沿发送数据；或基于所述I2C通信协议执行写操作时，在SCL信号的下降沿获取数据，基于所述I2C通信协议执行读操作时，在所述SCL信号的上升沿发送数据。需要说明的是，实现该协议的模块，能够实现无系统时钟，仅仅使用I2C的时钟进行工作。相比于依赖于系统主时钟的普通I2C协议，该模块的时钟仅仅为I2C的SCL时钟。在写操作时，在SCL的上升沿采集数据，在读操作时，在SCL的下降沿发送数据。也可以在写操作时，在SCL的下降沿采集数据，在读操作时，在SCL的上升沿发送数据，以实际应用场景为准，本申请实施例对此不做限定。

[0042]　在一实施例中，所述方法还包括：通过I2C接口模块与上位机和寄存器组控制器连接，其中，所述I2C接口模块设置在有限状态机上，所述I2C接口模块包括SCL接口、数据输入SDI接口、数据输出SDO接口、复位RST接口和控制器总线接口，所述寄存器组控制器配置为

控制I2C寄存器组、EE寄存器组以及EEPROM寄存器。

[0043] 图2是根据本申请实施例的一种可选的基于I2C总线异步批量读写寄存器的系统的结构示意图，设置在从机侧，如图2所示，所述系统包括：包含I2C接口模块的有限状态机(FSM)和寄存器组控制器。

[0044] 其中，总线状态检测模块接收SCL信号，并输出信号至有限状态机，有限状态机输出信号至输入输出模块，输入输出模块输出信号至总线状态检测模块，并与SDA信号通信连接，SDA信号包括SDI信号和SDO信号，其中，SDA信号为数据传输信号，SDI信号为数据输入信号，SDO信号为数据输出信号。

[0045] 寄存器组控制器内部包括读写信号检测模块、片内地址选择模块、I2C寄存器组、EEPROM控制器、EE寄存器组和输出多路选择器，其中，读写信号检测模块输出信号至片内地址选择模块，片内地址选择模块输出信号至I2C寄存器组，I2C寄存器组输出信号至输出多路选择器，EEPROM控制器输出信号至EE寄存器组，EE寄存器组输出信号至输出多路选择器，输出多路选择器输出模拟控制信号至模拟电路单元，EEPROM控制器通过EEPROM接口输出控制信号至EEPROM。

[0046] 在一实施例中，所述方法还包括：

[0047] 所述上位机通过所述SCL接口向所述有限状态机发送时钟信号；

[0048] 所述上位机通过所述SDI接口和所述SDO接口与所述有限状态机进行双向数据通信；以及

[0049] 所述上位机通过所述RST接口向所述有限状态机发送复位信号。

[0050] 如图2所示，在一实施例中，在所述控制器总线接口和所述寄存器组控制器之间设置通信总线，其中，所述通信总线配置为传输以下至少之一的数据信号：

[0051] 第一数据信号(EE_Address[7:0])，配置为指示所述上位机要访问的寄存器的地址；

[0052] 第二数据信号(Byte_CNT[6:0])，配置为指示一次传输的数据的Byte数；

[0053] 第三数据信号(rx_data[7:0])，配置为指示从I2C总线上接收到的8bit数据；

[0054] 第四数据信号(tx_data[7:0])，配置为指示将要通过I2C总线发送出去的8bit数据；

[0055] 第五数据信号(write_en)，配置为当需要写数据时，产生一个所述第五数据信号的脉冲；

[0056] 第六数据信号(read_en)，配置为当需要发送数据时，将所述第六数据信号拉高一个脉冲。

[0057] 图3是根据本申请实施例的一种可选的有限状态机的状态转移示意图，如图3所示，有限状态机基于所述I2C通信协议进行数据传输包括以下步骤：

[0058] S1，所述有限状态机在空闲状态时为IDLE状态，当检测到开始信号时，进入SALVE_ADDR状态，此时接收从机的地址；

[0059] S2，在接收8bit数据之后，所述有限状态机进入SLAVE_ACK状态，若所述上位机发送的地址与从机吻合，则应答ACK，之后根据所述I2C通信协议，进入WRITE_CMD(对应协议中的CMD控制字)状态，此状态用于接收当前操作为单Byte操作还是多Byte操作，以及接收当前操作的初始地址；

[0060] S3,在所述WRITE_CMD状态结束后,所述有限状态机进入SLAVE_ACK状态进行应答。

[0061] 需要说明的是,寄存器地址是从00h依次累加,为存储具体数据的地址。而从机地址为设备的地址,是上位机发起I2C传输需要的地址。

[0062] 在一实施例中,在上述步骤S3之后,所述方法还包括:

[0063] S4,若为单Byte操作,则首先进入WRITE_Byte操作,即为写单Byte状态,若在此时检测到开始信号,则进入READ_SLAVE_ADDR状态,继续接收一次从机地址,之后进入READ_Byte状态,读单Byte数据,之后进入MASTER_ACK,即等待上位机接收读数据之后给出应答以及停止信号,传输结束;

[0064] S5,若未检测到开始信号,则进行WRITE_Byte,写完数据之后进行SALVE_ACK,收到停止信号后传输结束;

[0065] S6,若为多Byte操作,则首先进入WRITE_CNT状态,若在此时检测到了开始信号,则进入READ_CNT状态,在读出CNT之后,继续读CNT个Byte结束传输;每次读完Byte之后进入MASTER_ACK,若当前完成Byte数小于READ_CNT阶段传输的CNT,则继续回到READ_BLOCK,若传输完CNT个Byte,回到IDLE;

[0066] S7,若未检测到开始信号,首先写WRITE_CNT,之后再写CNT个Byte之后结束传输;每次写完Byte之后进入SLAVE_ACK,若当前完成Byte数小于WRITE_CNT阶段传输的CNT,则继续回到WRITE_BLOCK,若传输完CNT个Byte,回到IDLE。

[0067] 图4是根据本申请实施例的一种可选的多时钟域同时配置模拟寄存器的原理图,如图4所示,该系统包含一个EEPROM,特定的EEPROM,相当于系统内部的存储器,在系统掉电的时候可以存储配置的值。I2C接口模块接收PAD_SDA信号、PAD_SCL信号,输出In_Enable信号、SDA_OUT信号,寄存器控制器(Reg_ctrl)中包含EE控制器(EE_ctrl)、EE寄存器组(ee_reg寄存器组)、I2C寄存器组(i2c_reg寄存器组)和多路选择器(MUX)。其中,PAD_SDA信号、PAD_SCL信号是I2C的SDA信号线和SCL信号线,分别是I2C的时钟和数据线,且SDA是一个双向信号线,其中上文中的SDO为输出,SDI为输入。In_Enable信号用来控制I2C端口SDA的方向,SDA_OUT相当于上文中提到的SDO。

[0068] 当晶振正常工作时,系统上电EE_ctrl默认读出EEPROM。

[0069] 在存在主时钟时能够通过主时钟控制进行读取EEPROM的值将其存放在ee_reg寄存器组(用于存储从EEPROM里读出的数据的寄存器)中,同时通过I2C能够修改的寄存器组记为i2c_reg,这两个寄存器组大小相同,分以下几种情况分别说明其使用方式。

[0070] 当存在主时钟,无I2C时钟时,通过主时钟将用户事先烧写配置好的EEPROM中的值读出,存放在ee_reg寄存器组之中,同时多路选择器选择ee_reg的值输出给模拟电路单元。

[0071] 当存在I2C时钟时,无论主时钟是否存在,首先将ee_reg的值赋值给i2c_reg,这样I2C协议就在ee_reg的值的基础上对模拟进行配置,同时输出多路选择器选择i2c_reg的值输出给模拟电路单元。

[0072] 通过I2C进行直接修改的寄存器为i2c_reg。若通过I2c协议读出寄存器的值,则读的值取自于Reg_I2C。

[0073] 若无I2C的时钟,则MUX输出给模拟的数据来自于ee_reg,若有I2C的时钟,则MUX输出给模拟的数据来自于Reg_I2C。MUX的选择取决于是否进行过I2C的通信。

[0074] 通过本申请提供的实施例,首先解决了一个当前I2C协议传输效率相对低下的问

题，能够实现批量传输数据，其次解决的问题是当前通过I2C配置寄存器组普遍需要主时钟的参与，同时芯片配置寄存器组大多使用同步设计，兼容性不强，当系统时钟出现问题时就不能对寄存器组进行配置，当前方案解决了当无系统时钟时能够进行配置寄存器组，同时兼容了当有时钟时能够通过写EEPROM进行存储配置，以及将EEPROM中的数据载入到寄存器组。

[0075] 上述本申请实施例序号仅仅为了描述，不代表实施例的优劣。

[0076] 在本申请的上述实施例中，对各个实施例的描述都各有侧重，某个实施例中没有详述的部分，可以参见其他实施例的相关描述。

[0077] 以上所述仅是本申请的优选实施方式，应当指出，对于本技术领域的普通技术人员来说，在不脱离本申请原理的前提下，还可以做出若干改进和润饰，这些改进和润饰也应视为本申请的保护范围。


在I2C通信协议中设置命令指示符CMD控制字和EE_Address控制字
S102
在一次传输中需要发送多个Byte数据时，以所述EE_Address控制字为起始地址依次连续发送所述多个Byte数据
S104
通过I2C接口模块与上位机和寄存器组控制器连接，通过有限状态机基于所述I2C通信协议进行数据传输，
S106

图 1

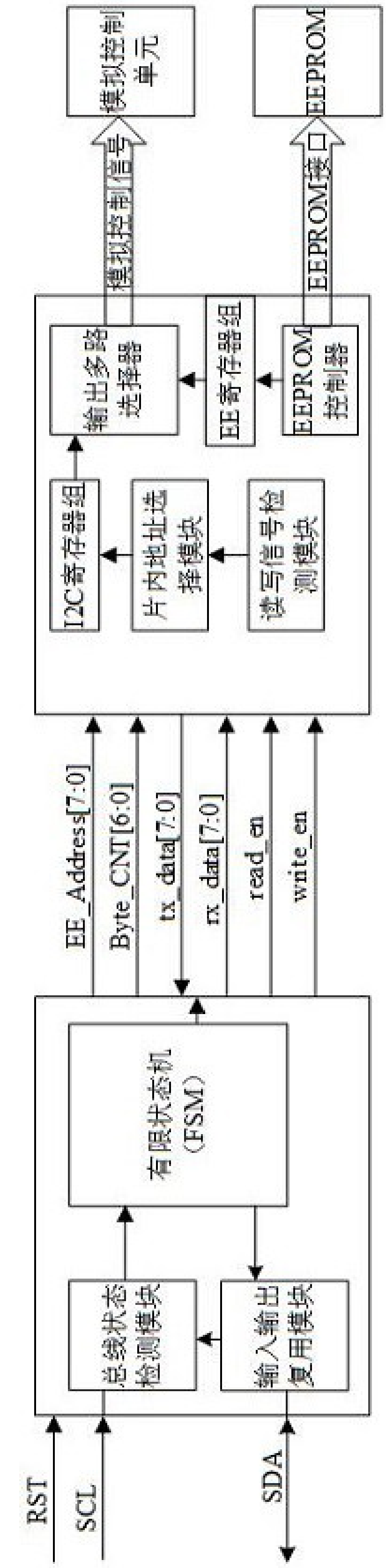

模拟控制单元
EEPROM
模拟控制信号
EEPROM接口
输出多路选择器
EE寄存器组
EEPROM控制器
I2C寄存器组
片内地址选择模块
读写信号检测模块
EE_Address[7:0]
Byte_CNT[6:0]
tx_data[7:0]
rx_data[7:0]
read_en
write_en
有限状态机（FSM）
总线状态检测模块
输入输出复用模块
RST
SCL
SDA

图 2

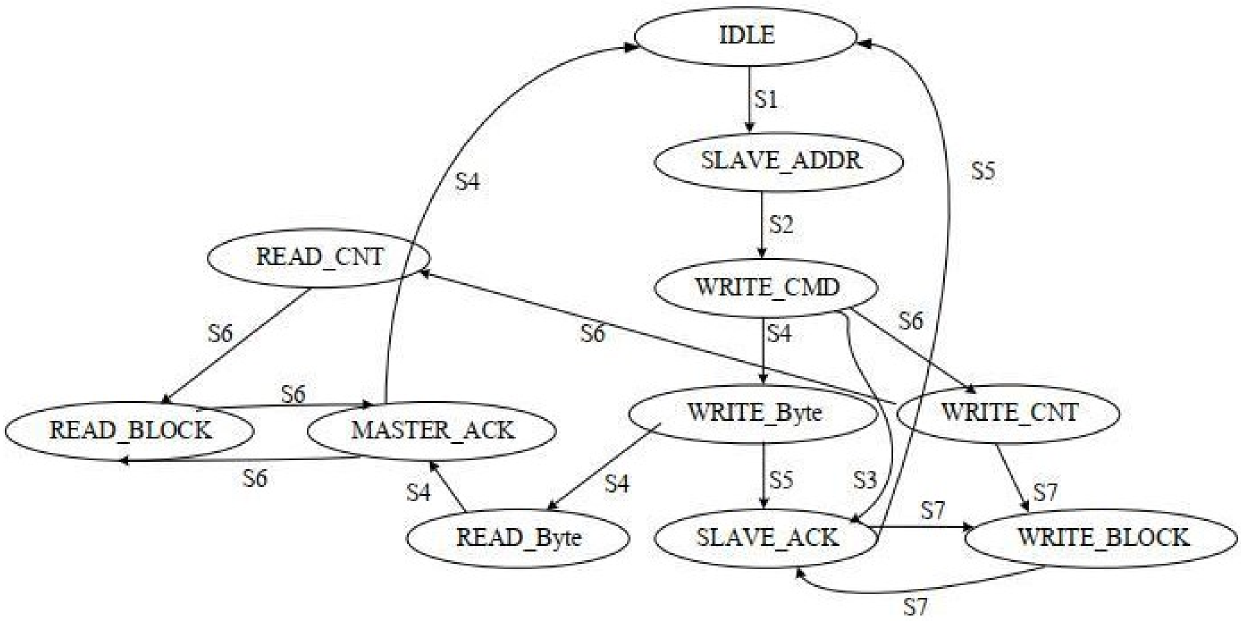

IDLE
S1
SLAVE_ADDR
S2
WRITE_CMD
S4
S6
READ_CNT
S6
S6
S4
S5
READ_BLOCK
S6
MASTER_ACK
S6
WRITE_Byte
WRITE_CNT
S4
S4
S5
S3
S7
READ_Byte
SLAVE_ACK
S7
WRITE_BLOCK
S7

图 3

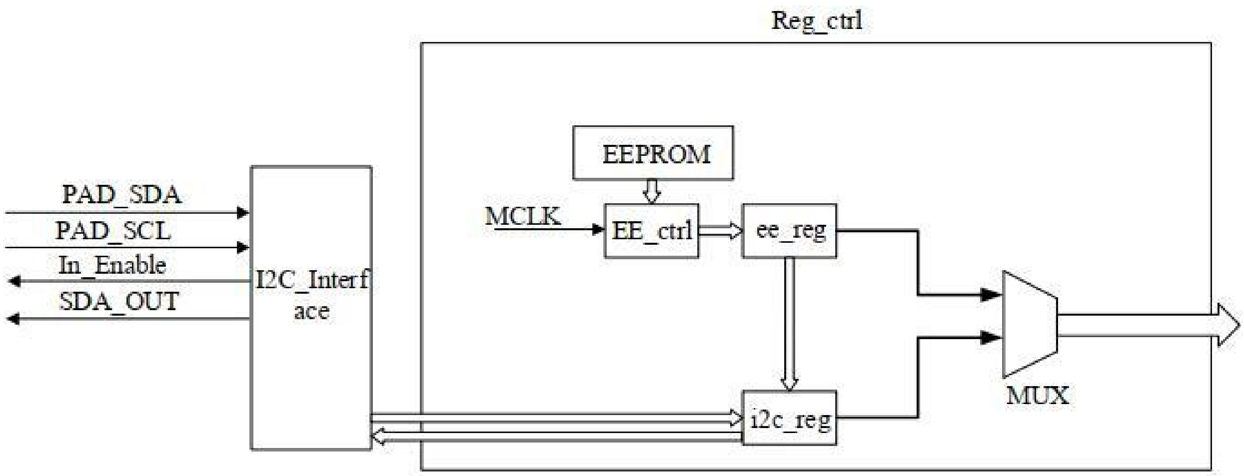

Reg_ctrl
EEPROM
PAD_SDA
PAD_SCL
In_Enable
SDA_OUT
I2C_Interface
MCLK
EE_ctrl
ee_reg
i2c_reg
MUX

图 4