



(21)申请号 201510885023.X

(22)申请日 2015.12.04

(65)同一申请的已公布的文献号

申请公布号 CN 105549552 A

(43)申请公布日 2016.05.04

(73)专利权人 中国科学院苏州生物医学工程技术研究所

地址 215000 江苏省苏州市苏州高新区科
灵路88号

(72)发明人 周哲 董月芳 陈爽 付威威

(74)专利代理机构 北京路浩知识产权代理有限公司 11002

代理人 李相雨 李官

(51)Int.Cl.

G05B 19/418(2006.01)

(56)对比文件

CN 202602678 U,2012.12.12,

CN 203632688 U,2014.06.04,

CN 204423250 U,2015.06.24,

WO 2005066730 A2,2005.07.21,

EP 1702246 B1,2015.06.03,

曲波 等.基于CPLD的多通道CAN总线接口设计.《仪器仪表学报》.2010,第31卷(第8期),

曲波 等.基于CPLD的多通道CAN总线接口设计.《仪器仪表学报》.2010,第31卷(第8期),

审查员 徐倩茹

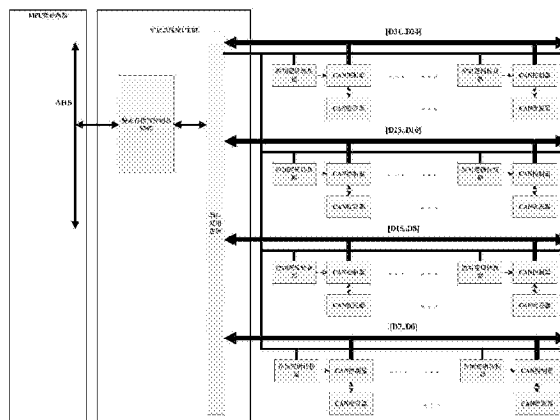
权利要求书2页 说明书7页 附图6页

(54)发明名称

基于Linux的CAN总线扩展系统及方法

(57)摘要

本发明涉及一种基于Linux的CAN总线扩展系统和方法,包括:MPU微处理器、扩展总线接口EBI、控制逻辑转换器、总线驱动器、CAN控制器、CAN收发器;MPU访问CAN控制器;EBI产生数据总线信号、地址总线信号和第一控制逻辑信号,将数据总线信号、地址总线信号和第一控制逻辑信号传输至I/O端口上;控制逻辑转换器将地址总线信号和第一控制逻辑信号转换成访问CAN控制器所需的第二控制逻辑信号;总线驱动器在第一控制逻辑信号的控制下对数据总线信号传输方向进行控制;CAN控制器在第二控制逻辑信号和总线驱动器共同驱动下,实现与CAN收发器的接收与发送。本发明数据访问速度快,出错几率小,可以同时驱动多个CAN控制器并行工作,摆脱了计算机体积、电源的限制。



1. 一种基于Linux的CAN总线扩展系统,其特征在于,包括:

MPU微处理器、扩展总线接口EBI、控制逻辑转换器、总线驱动器、CAN控制器、CAN收发器;

所述MPU微处理器用于访问所述CAN控制器中的寄存器;

所述扩展总线接口EBI用于在访问所述寄存器时产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上;

所述控制逻辑转换器用于根据所述地址总线信号和第一控制逻辑信号产生访问所述CAN控制器所需的第二控制逻辑信号;

所述总线驱动器用于在所述第一控制逻辑信号的控制下使能与控制所述数据总线信号的传输方向;

所述CAN控制器用于在所述第二控制逻辑信号和所述总线驱动器的驱动下,实现所述CAN控制器的寄存器访问,在所述MPU微处理器的控制下,实现与所述CAN收发器的接收与发送;

其中,所述控制逻辑转换器的控制时序为:

$$\overline{ALE} = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0} + \overline{NWE} + \overline{NCS}$$

$$\overline{WR} = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0} + \overline{NWE} + \overline{NCS} \text{ 或者 } \overline{WR} = \overline{ADD0} + \overline{NWE}$$

$$\overline{RD} = \overline{ADD0} \& \overline{NRD} \& \overline{NCS} = \overline{ADD0} + \overline{NRD} + \overline{NCS} \text{ 或者 } \overline{RD} = \overline{ADD0} + \overline{NRD}$$

$$\overline{CS} = \overline{ADD0} \& \overline{NCS} = \overline{ADD0} + \overline{NCS}$$

其中,ALE表示第二控制逻辑信号中的地址锁存允许信号,RD表示第二控制逻辑信号中的读信号,WR表示第二控制逻辑信号中的写信号,CS表示第二控制逻辑信号中的片选信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,ADD0表示地址总线信号中的其中一个地址信号,NWE表示第一控制逻辑信号中的写信号。

2. 根据权利要求1所述的基于Linux的CAN总线扩展系统,其特征在于,所述处理器包括高性能外设总线AHB;所述MPU微处理器用于通过所述高性能外设总线AHB访问所述CAN控制器中的寄存器。

3. 根据权利要求1所述的基于Linux的CAN总线扩展系统,其特征在于,所述扩展总线接口EBI包括静态存储器控制器SMC和端口复用逻辑;

所述静态存储器控制器SMC用于在访问所述寄存器时产生第一数据总线信号、地址总线信号和第一控制逻辑信号;并根据端口复用逻辑将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上。

4. 根据权利要求1-3任一所述的基于Linux的CAN总线扩展系统,其特征在于,所述总线驱动器的一端通过数据总线连接所述扩展总线接口EBI,另一端通过地址/数据复用总线连接所述CAN控制器。

5. 根据权利要求4所述的基于Linux的CAN总线扩展系统,所述总线驱动器的控制时序为:

$$\overline{AD[7:0]} = \overline{D[31:24]} \& \overline{NCS}$$

DIR=NRD CS₁=NCS

其中,AD[7:0]表示地址/数据复用总线的输出信号,D[31:24]表示数据总线的输入信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,CS₁所示总线驱动器的片选信号,DIR表示总线驱动器的方向信号。

6.一种基于Linux的CAN总线扩展方法,其特征在于,包括:

通过MPU微处理器访问CAN控制器中的寄存器;

在访问所述寄存器时,通过扩展总线接口EBI产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口EBI的对应输入/输出端口上;

根据所述地址总线信号和第一控制逻辑信号,通过控制逻辑转换器产生访问所述CAN控制器所需的第二控制逻辑信号;

在所述第一控制逻辑信号的控制下,通过总线驱动器使能与控制所述数据总线信号的传输方向;

在所述第二控制逻辑信号和所述总线驱动器的驱动下,实现所述CAN控制器的寄存器访问,在所述MPU微处理器的控制下,实现CAN控制器与CAN收发器的接收与发送;

其中,所述控制逻辑转换器的控制时序为:

$$\overline{ALE} = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0} + \overline{NWE} + \overline{NCS}$$

$$\overline{WR} = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0} + \overline{NWE} + \overline{NCS} \text{ 或者 } \overline{WR} = \overline{ADD0} + \overline{NWE}$$

$$\overline{RD} = \overline{ADD0} \& \overline{NRD} \& \overline{NCS} = \overline{ADD0} + \overline{NRD} + \overline{NCS} \text{ 或者 } \overline{RD} = \overline{ADD0} + \overline{NRD}$$

$$\overline{CS} = \overline{ADD0} \& \overline{NCS} = \overline{ADD0} + \overline{NCS}$$

其中,ALE表示第二控制逻辑信号中的地址锁存允许信号,RD表示第二控制逻辑信号中的读信号,WR表示第二控制逻辑信号中的写信号,CS表示第二控制逻辑信号中的片选信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,ADD0表示地址总线信号中的其中一个地址信号,NWE表示第一控制逻辑信号中的写信号。

7.根据权利要求6所述的基于Linux的CAN总线扩展方法,其特征在于,所述通过MPU微处理器访问所述CAN控制器中的寄存器具体包括:

通过所述MPU微处理器中的高性能外设总线AHB访问所述CAN控制器中的寄存器。

8.根据权利要求7所述的基于Linux的CAN总线扩展方法,其特征在于,

所述在访问所述寄存器时,通过扩展总线接口EBI产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口EBI的对应输入/输出端口上具体包括:

在访问所述寄存器时,通过所述扩展总线接口EBI的静态存储器控制器SMC产生第一数据总线信号、地址总线信号和第一控制逻辑信号;并根据端口复用逻辑将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上。

9.根据权利要求6所述的基于Linux的CAN总线扩展方法,其特征在于,该方法还包括:所述MPU微处理器通过所述扩展总线接口EBI挂载4组CAN控制器和CAN收发器,每组挂载多路CAN控制器和CAN收发器。

基于Linux的CAN总线扩展系统及方法

技术领域

[0001] 本发明涉及电子技术领域,特别涉及一种基于Linux的CAN总线扩展系统及方法。

背景技术

[0002] CAN是控制器局域网(Controllor Area Network,CAN)的简称,是由研发和生产电子产品著称的德国BOSCH公司开发的,并最终成为国际标准(ISO 11898),是国际上应用最广泛的现场总线之一。在北美和西欧,CAN总线协议已经成为汽车计算机控制系统和嵌入式工业控制局域网的标准总线。CAN的高性能和可靠性已被认同,并被广泛地应用于工业自动化、船舶、医疗设备、工业设备等方面。现场总线是当今自动化领域技术发展的热点之一,被誉为自动化领域的计算机局域网。它的出现为分布式控制系统实现各节点之间实时、可靠的数据通信提供了强有力的技术支持。

[0003] 一个可用的CAN网络节点包括CAN控制器与CAN收发器,CAN收发器用于物理层信号的接受、发送,将单端信号与CAN总线上的差分信号互相转换,现有产品如MCP2551、TJA1050、TJA1040等;CAN控制器用于CAN网络的波特率设置、报文滤波设置、报文接受与发送、传输错误等的控制。现有的大部分32位的ARM微处理器MCU内部集成了CAN控制器,例如STM32系列、LM3S系列、LPC系列单片机,还有外置的如MCP2515、SJA1000等。

[0004] 现有技术方案一:如图1所示,采用单个ARM处理器,内部集成CAN控制器、USB接口,一端通过CAN收发器与CAN总线相连,另一端通过USB虚拟串口(VCP)与上位机USB接口相连,只需要将虚拟串口数据或指令通过ARM处理器转换成CAN控制器的配置与报文的收发处理。

[0005] 现有技术方案二:如图2所示,采用单个ARM、MSP430或者8051处理器,内部集成SPI控制器、USB接口,一端通过SPI接口与外置的CAN控制器相连,CAN控制器通过收发器与CAN总线相连,另一端通过USB虚拟串口(VCP)与上位机USB接口相连。

[0006] 现有技术方案三:如图3所示,采用单个MSP430或者8051处理器,具有带ALE(Address Latch Enable)地址锁存的外部扩展总线、USB接口,一端通过外部总线接口与外置并行CAN控制器相连,CAN控制器通过收发器与CAN总线相连,另一端通过USB虚拟串口(VCP)与上位机USB接口相连。

[0007] 传统的个人计算机(PC)实现对CAN总线分析与调试,一般通过以下三种技术手段:一、使用PCI-Express(PCI-E)接口的CAN总线协议采集卡;二、采用USB转CAN调试器,三、CAN总线分析仪,例如致远电子的CANScope总线协议分析仪。

[0008] 无论采用PCI-E的CAN总线采集卡、USB转CAN或者CAN分析仪,设备或者装置均无法脱离计算机、笔记本单独使用,这在很多工业现场应用环境下,不具备这样的体积空间与电源要求;采用现有的技术,均是有所取舍,或者造价昂贵、功能复杂,或者价格低廉性能降低,效率低下,出错概率高;采用现有的技术方案,CAN总线的扩展性能均比较差,只能同时调试单个/两个CAN网络。

发明内容

[0009] 本发明所要解决的技术问题是如何提高CAN总线访问效率,减少出错几率。

[0010] 为此目的,本发明提出了一种基于Linux的CAN总线扩展系统,包括:

[0011] MPU微处理器、扩展总线接口EBI、控制逻辑转换器、总线驱动器、CAN控制器、CAN收发器;

[0012] 所述MPU微处理器用于访问所述CAN控制器中的寄存器;

[0013] 所述扩展总线接口EBI用于在访问所述寄存器时产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上;

[0014] 所述控制逻辑转换器用于根据所述地址总线信号和第一控制逻辑信号产生访问所述CAN控制器所需的第二控制逻辑信号;

[0015] 所述总线驱动器用于在所述第一控制逻辑信号的控制下使能与控制所述数据总线信号的传输方向;

[0016] 所述CAN控制器用于在所述第二控制逻辑信号和所述总线驱动器的驱动下,实现所述CAN控制器的寄存器访问,在所述MPU微处理器的控制下,实现与所述CAN收发器的接收与发送。

[0017] 优选地,所述处理器包括高性能外设总线AHB;所述MPU微处理器用于通过所述高性能外设总线AHB访问所述CAN控制器中的寄存器。

[0018] 优选地,所述扩展总线接口EBI包括静态存储器控制器SMC和端口复用逻辑;

[0019] 所述静态存储器控制器SMC用于在访问所述寄存器时产生第一数据总线信号、地址总线信号和第一控制逻辑信号;并根据端口复用逻辑将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上。

[0020] 优选地,所述总线驱动器的一端通过数据总线连接所述扩展总线接口EBI,另一端通过地址/数据复用总线连接所述CAN控制器。

[0021] 优选地,所述总线驱动器的控制时序为:

[0022] $AD[7:0] = D[31:24] \& \overline{NCS}$

[0023] $DIR = \overline{NRD} \quad CS_1 = \overline{NCS}$

[0024] 其中,AD[7:0]表示地址/数据复用总线的输出信号,D[31:24]表示数据总线的输入信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,CS₁所示总线驱动器的片选信号,DIR表示总线驱动器的方向信号。

[0025] 优选地,所述控制逻辑转换器的控制时序为:

[0026] $ALE = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0 + NWE + NCS}$

[0027] $WR = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0 + NWE + NCS}$ 或者 $WR = ADD0 + NWE$

[0028] $RD = \overline{ADD0} \& \overline{NRD} \& \overline{NCS} = \overline{ADD0 + NRD + NCS}$ 或者 $RD = ADD0 + NRD$

[0029] $CS = \overline{ADD0} \& \overline{NCS} = \overline{ADD0 + NCS}$

[0030] 其中,ALE表示第二控制逻辑信号中的地址锁存允许信号,RD表示第二控制逻辑信号中的读信号,WR表示第二控制逻辑信号中的写信号,CS表示第二控制逻辑信号中的片选信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,

ADD0表示地址总线信号中的其中一个地址信号,NWE表示第一控制逻辑信号中的写信号。

[0031] 另一方面,采用上述的基于Linux的CAN总线扩展系统,本发明还提供了一种基于Linux的CAN总线扩展方法,包括:

[0032] 通过MPU微处理器访问CAN控制器中的寄存器;

[0033] 在访问所述寄存器时,通过扩展总线接口EBI产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口EBI的对应输入/输出端口上;

[0034] 根据所述地址总线信号和第一控制逻辑信号,通过控制逻辑转换器产生访问所述CAN控制器所需的第二控制逻辑信号;

[0035] 在所述第一控制逻辑信号的控制下,通过总线驱动器使能与控制所述数据总线信号的传输方向;

[0036] 在所述第二控制逻辑信号和所述总线驱动器的驱动下,实现所述CAN控制器的寄存器访问,在所述MPU微处理器的控制下,实现CAN控制器与CAN收发器的接收与发送。

[0037] 优选地,所述通过MPU微处理器访问所述CAN控制器中的寄存器具体包括:

[0038] 通过所述MPU微处理器中的高性能外设总线AHB访问所述CAN控制器中的寄存器。

[0039] 优选地,所述在访问所述寄存器时,通过扩展总线接口EBI产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口EBI的对应输入/输出端口上具体包括:

[0040] 在访问所述寄存器时,通过所述扩展总线接口EBI的静态存储器控制器SMC产生第一数据总线信号、地址总线信号和第一控制逻辑信号;并根据端口复用逻辑将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上。

[0041] 优选地,该方法还包括:所述MPU微处理器通过所述扩展总线接口EBI挂载4组CAN控制器和CAN收发器,每组挂载多路CAN控制器和CAN收发器。

[0042] 通过采用本发明所提供的基于Linux的CAN总线扩展系统及方法,高性能MPU通过EBI总线直接扩展CAN总线,高性能MPU的EBI并行总线访问速度快,中间仅需要硬件信号转换匹配,没有软件协议转换过程,减少了出错几率,提高了CAN控制器的性能与效率;采用EBI总线扩展的方式,由于EBI总线本身扩展性非常强,所以采用该种方式可以扩展多路CAN总线,可以同时驱动多个CAN控制器并行工作。通过基于Linux开源平台实现,平台稳定性高,既提高了性能,又可以保证技术的稳定、可靠性,同时降低了研发生产成本;本发明完全可以集成在一台单独的小型便携式、手持式设备上开发与使用,摆脱了计算机体积、电源的限制,大大提高了现场使用的灵活性。

附图说明

[0043] 通过参考附图会更加清楚的理解本发明的特征和优点,附图是示意性的而不应该理解为对本发明进行任何限制,在附图中:

[0044] 图1-3示出了现有技术中对CAN总线的分析与调试的框架结构示意图;

[0045] 图4示出了本发明MPU通过EBI总线连接CAN控制器的扩展结构示意图;

[0046] 图5示出了本发明MPU通过EBI总线连接单个CAN控制器的示例图;

[0047] 图6示出了本发明静态存储器控制器在AHB进行读取操作时产生的控制逻辑信号时序示意图；

[0048] 图7示出了本发明静态存储器控制器在AHB进行写入操作时产生的控制逻辑信号时序示意图；

[0049] 图8示出了本发明对CAN控制器进行读取操作时CAN控制器所需的控制逻辑信号时序示意图；

[0050] 图9示出了本发明对CAN控制器进行写入操作时CAN控制器所需的控制逻辑信号时序示意图；

[0051] 图10示出了本发明在读取操作时，静态存储器控制器总线周期扩展时序与CAN控制器延展时序对比示意图；

[0052] 图11示出了本发明在写入操作时，静态存储器控制器总线周期扩展时序与CAN控制器延展时序对比示意图；

[0053] 图12示出了本发明控制逻辑转换器进行逻辑转换的示意图；

[0054] 图13示出了在本发明的基础上基于Socket的CAN控制器网络设备驱动以及CAN控制器应用程序调试分析方法示意图。

具体实施方式

[0055] 下面将结合附图对本发明的实施例进行详细描述。

[0056] 如图4、5、12所示，本发明提供了一种基于Linux的CAN总线扩展系统，包括：MPU微处理器、扩展总线接口EBI、控制逻辑转换器、总线驱动器、CAN控制器、CAN收发器；

[0057] 所述MPU微处理器用于访问所述CAN控制器中的寄存器；优选地，所述处理器包括高性能外设总线AHB；所述MPU微处理器用于通过所述高性能外设总线AHB访问所述CAN控制器中的寄存器。

[0058] 所述扩展总线接口EBI用于在访问所述寄存器时产生数据总线信号、地址总线信号和第一控制逻辑信号；并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上；优选地，所述扩展总线接口EBI包括静态存储器控制器SMC和端口复用逻辑；所述静态存储器控制器SMC用于在访问所述寄存器时产生第一数据总线信号、地址总线信号和第一控制逻辑信号；并根据端口复用逻辑将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上。

[0059] 所述控制逻辑转换器用于根据所述地址总线信号和第一控制逻辑信号产生访问所述CAN控制器所需的第二控制逻辑信号；

[0060] 所述总线驱动器用于在所述第一控制逻辑信号的控制下使能与控制所述数据总线信号的传输方向；

[0061] 所述CAN控制器用于在所述第二控制逻辑信号和所述总线驱动器的驱动下，实现所述CAN控制器的寄存器访问，在所述MPU微处理器的控制下，实现与所述CAN收发器的接收与发送。

[0062] 由于高性能MPU有32位数据总线带宽，而一般的并行CAN控制器只有8位数据，因此每8位数据总线均可挂载CAN控制器与CAN收发器，而CAN控制器的控制逻辑信号由统一的

SMC产生的控制逻辑信号经过一个控制逻辑转换器形成。如果SMC产生的控制逻辑信号越多,则通过每一个控制逻辑转换器产生独立的CAN控制器的控制逻辑信号也越多,因此可以扩展多路CAN总线。

[0063] 如图5所示,为了进一步说明扩展的逻辑过程,将其中一个扩展CAN控制器分立出来,进行详细说明。高性能外设总线AHB访问CAN控制器的寄存器时,静态存储器控制器SMC将产生总线信号,包括数据总线D[31:0]、地址总线A[25:0]、控制总线(NCS、NRD、NWR、NBS0、NBS1、NBS2、NBS3)等信号。端口复用逻辑用于选通输入输出(I/O)端口的功能,因为MPU的管脚大部分是复用的,每个I/O端口除了可以分配给静态存储器控制器使用,也可以分配给其他模块使用,或者作为通用输入输出端口(GPIO)。也就是说静态存储器控制器通过端口复用逻辑将所需要的地址总线、数据总线、控制总线向外部(I/O)端口输出。端口复用逻辑通过数据总线D[31:0]与CAN控制器的数据/地址复用总线AD[7:0]相连接,端口复用逻辑可以输出32位数据总线D[31:0],分成4路D[31:24]、D[23:16]、D[15:8]、D[7:0],每路8位数据(例如D[31:24])与CAN控制器的数据/地址总线AD[7:0]相连接。而CAN控制器所需要的控制信号(ALE、WR、RD、CS),则由端口复用逻辑输出的地址总线A[25:0]、控制总线(NCS、NRD、NWR、NBS0、NBS1、NBS2、NBS3)经过控制逻辑转换器变换后产生。CAN控制器通过接受、发送信号与CAN收发器相连接,CAN收发器通过高速差分信号与CAN总线相连接。本发明中最重要的部分就是根据静态存储器控制器产生的第一控制逻辑信号,以及访问CAN控制器所需要的第二控制逻辑信号,设计控制逻辑转换将两者的控制逻辑信号进行匹配。

[0064] 如图6、7所示,为静态存储器控制器产生的第一控制逻辑信号,图6为静态存储器控制器在高性能外设总线对CAN控制器的寄存器进行读取操作时产生的第一控制逻辑时序,图7为在高性能外设总线对CAN控制器的寄存器进行写入操作时产生的第一控制逻辑时序。

[0065] 如图8、9所示,为CAN控制器所需的第二控制逻辑信号,图8为对CAN控制器进行读取操作时所需的第二控制逻辑时序,图9为对CAN控制器进行写入操作时所需的第二控制逻辑时序。

[0066] 对比图8、9所示CAN控制器所需的第二控制逻辑信号与图6、7所示静态存储器控制器产生的第一控制逻辑信号,不难看出静态存储器控制器必须进行两个总线周期操作才能产生CAN控制器所需的第二控制逻辑信号,因此将静态存储器控制器进行两个总线周期操作扩展。

[0067] 如图10、图11所示,将静态存储器控制器进行两个总线周期操作扩展,并将CAN控制器所需的第二控制逻辑信号,地址锁存总线操作与数据读取、写入总线操作之间的时间间隔加长,并从A[25:0]地址总线中任意引入一根地址线,对地址锁存操作周期与数据读取、写入周期加以区分控制。假设由A[25:0]地址总线中任意引入一根地址线为ADD0,输入数据总线信号为D[31:0](取其31:24位),控制逻辑转换器输入信号为A[25:0](取其ADD0)、NWE、NRD、NCS,输出信号为AD[7:0]、ALE、WR、RD、CS。

[0068] 由以上的时序分析可以得到输入信号与输出信号的逻辑关系,首先,输出地址数据总线AD[7:0]信号应该由输入数据总线信号D[31:24]与输入控制信号NCS组合得到,由于输入信号D[31:24]与输出信号AD[7:0]均为8位宽的总线信号,为此在控制逻辑转换中引入总线驱动器,所述总线驱动器的一端通过数据总线连接所述扩展总线接口EBI,另一端通过

地址/数据复用总线连接所述CAN控制器。总线驱动器的CS信号由输入信号NCS产生,总线驱动器通过DIR信号控制数据的方向,DIR信号取决于输入信号NRD,由于两个总线周期扩展操作,第一个周期无论读写总是写入地址,即NWE在第一个周期均置低,第二个周期根据读写时序才会产生总线信号输入或者输出,因此DIR信号取决于输入信号NRD。其次,其他控制输出信号(ALE、WR、RD、CS)应该由输入地址信号A[25:0](取其一ADD0)与其他输入控制信号NWE、NRD、NCS经过逻辑组合得到。从以上时序可得控制逻辑转换器逻辑实现如下,控制逻辑转换器逻辑实现与控制示意图如图12所示。

[0069] 其中,总线驱动器控制时序为:

[0070] $AD[7:0] = D[31:24] \& \overline{NCS}$

[0071] DIR=NRD CS=NCS

[0072] 其中,AD[7:0]表示地址/数据复用总线的输出信号,D[31:24]表示数据总线的输入信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,CS₁所示总线驱动器的片选信号,DIR表示总线驱动器的方向信号。

[0073] 其中,控制逻辑转换器控制时序为:

[0074] $ALE = ADD0 \& \overline{NWE} \& \overline{NCS} = \overline{ADD0 + NWE + NCS}$

[0075] $WR = \overline{ADD0} \& \overline{NWE} \& \overline{NCS} = \overline{ADD0 + NWE + NCS}$ 或者 WR=ADD0+NWE

[0076] $RD = \overline{ADD0} \& \overline{NRD} \& \overline{NCS} = \overline{ADD0 + NRD + NCS}$ 或者 RD=ADD0+NRD

[0077] $CS = \overline{ADD0} \& \overline{NCS} = \overline{ADD0 + NCS}$

[0078] 其中,ALE表示第二控制逻辑信号中的地址锁存允许信号,RD表示第二控制逻辑信号中的读信号,WR表示第二控制逻辑信号中的写信号,CS表示第二控制逻辑信号中的片选信号,NCS表示第一控制逻辑信号中的片选信号,NRD表示第一控制逻辑信号中的读信号,ADD0表示地址总线信号中的其中一个地址信号,NWE表示第一控制逻辑信号中的写信号。

[0079] 另一方面,本发明还提供了一种基于Linux的CAN总线扩展方法,包括:

[0080] 通过MPU微处理器访问CAN控制器中的寄存器;

[0081] 在访问所述寄存器时,通过扩展总线接口EBI产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口EBI的对应输入/输出端口上;

[0082] 根据所述地址总线信号和第一控制逻辑信号,通过控制逻辑转换器产生访问所述CAN控制器所需的第二控制逻辑信号;

[0083] 在所述第一控制逻辑信号的控制下,通过总线驱动器使能与控制所述数据总线信号的传输方向;

[0084] 在所述第二控制逻辑信号和所述总线驱动器的驱动下,实现所述CAN控制器的寄存器访问,在所述MPU微处理器的控制下,实现CAN控制器与CAN收发器的接收与发送。

[0085] 其中较优地,所述通过MPU微处理器访问所述CAN控制器中的寄存器具体包括:

[0086] 通过所述MPU微处理器中的高性能外设总线AHB访问所述CAN控制器中的寄存器。

[0087] 其中较优地,所述在访问所述寄存器时,通过扩展总线接口EBI产生数据总线信号、地址总线信号和第一控制逻辑信号;并将所述数据总线信号、地址总线信号和第一控制

逻辑信号传输至所述扩展总线接口EBI的对应输入/输出端口上具体包括:

[0088] 在访问所述寄存器时,通过所述扩展总线接口EBI的静态存储器控制器SMC产生第一数据总线信号、地址总线信号和第一控制逻辑信号;并根据端口复用逻辑将所述数据总线信号、地址总线信号和第一控制逻辑信号传输至所述扩展总线接口的对应输入/输出端口上。另外,所述MPU微处理器可以通过所述扩展总线接口EBI挂载4组CAN控制器和CAN收发器,每组挂载多路CAN控制器和CAN收发器。

[0089] 另外,本发明可以采用基于Linux的CAN总线分析与调试方法,上述EBI扩展CAN总线意在通过总线驱动器以及控制逻辑转换器,将CAN控制器作为一种外部存储设备连接到EBI内部的静态存储器控制器SMC上,这样MPU就可以像访问内部存储器一样通过AHB总线直接对CAN控制器进行访问控制。如图13所示,在Linux操作系统下,基于套接字(Socket)的CAN网络设备驱动以及CAN应用程序分析调试方法如下,在软件上可以将CAN控制器作为外部存储设备进行驱动,将CAN控制器驱动添加到Linux内核中,Linux内核中CAN设备被作为一种网络设备添加到Linux网络设备驱动层,对Linux网络设备的访问一般采用基于Socket的方式,对CAN网络而言Linux内核提供了基于Socket的CAN接口层,再向上是应用层,应用层通过网际协议路由策略II(ip[route2])应用程序对CAN网络设备进行配置,包括设置波特率、采样率、工作模式、位时隙参数、设备启动/停止等,对CAN报文的收发采用基于CAN套接字应用程序库(libsocketcan程序库)的CAN通信测试工具(canutils应用程序),libsocketcan程序库是基于Socket方式对CAN网络设备进行基本应用的支持库,在该程序库的支持下canutils应用程序可以进行CAN报文的接收与发送,而在通信过程中CAN网络设备产生的总线错误、传输错误,在Linux系统中会以网络设备日志的方式进行统计,通过查看网络设备连接状态可以知道CAN总线上产生的各种错误。

[0090] 本发明设计了一种基于Linux的CAN总线扩展系统及方法,高性能MPU通过EBI总线直接扩展CAN总线,高性能MPU的EBI并行总线访问速度快,中间仅需要硬件信号转换匹配,没有软件协议转换过程,减少了出错几率,提高了CAN控制器的性能与效率;采用EBI总线扩展的方式,由于EBI总线本身扩展性非常强,所以采用该种方式可以扩展多路CAN总线,可以同时驱动多个CAN控制器并行工作。通过基于Linux开源平台实现,平台稳定性高,既提高了性能,又可以保证技术的稳定、可靠性,同时降低了研发生产成本;本发明完全可以集成在一台单独的小型便携式、手持式设备上开发与使用,摆脱了计算机体积、电源的限制,大大提高了现场使用的灵活性。

[0091] 虽然结合附图描述了本发明的实施方式,但是本领域技术人员可以在不脱离本发明的精神和范围的情况下做出各种修改和变型,这样的修改和变型均落入由所附权利要求所限定的范围之内。

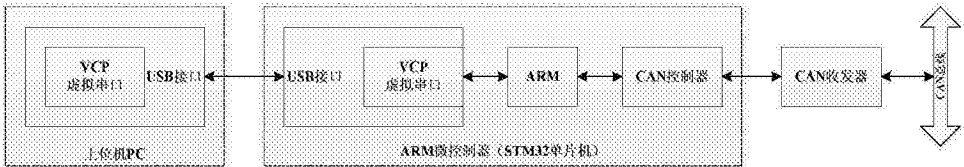


图1

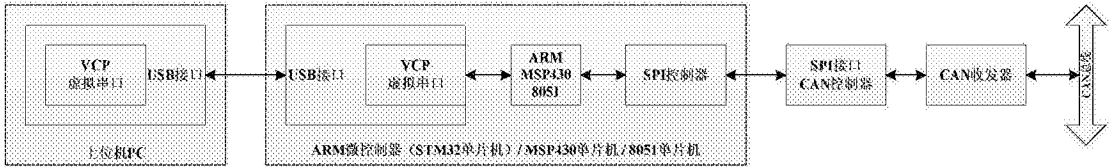


图2

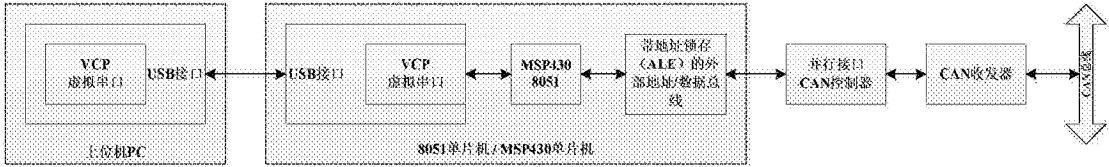


图3

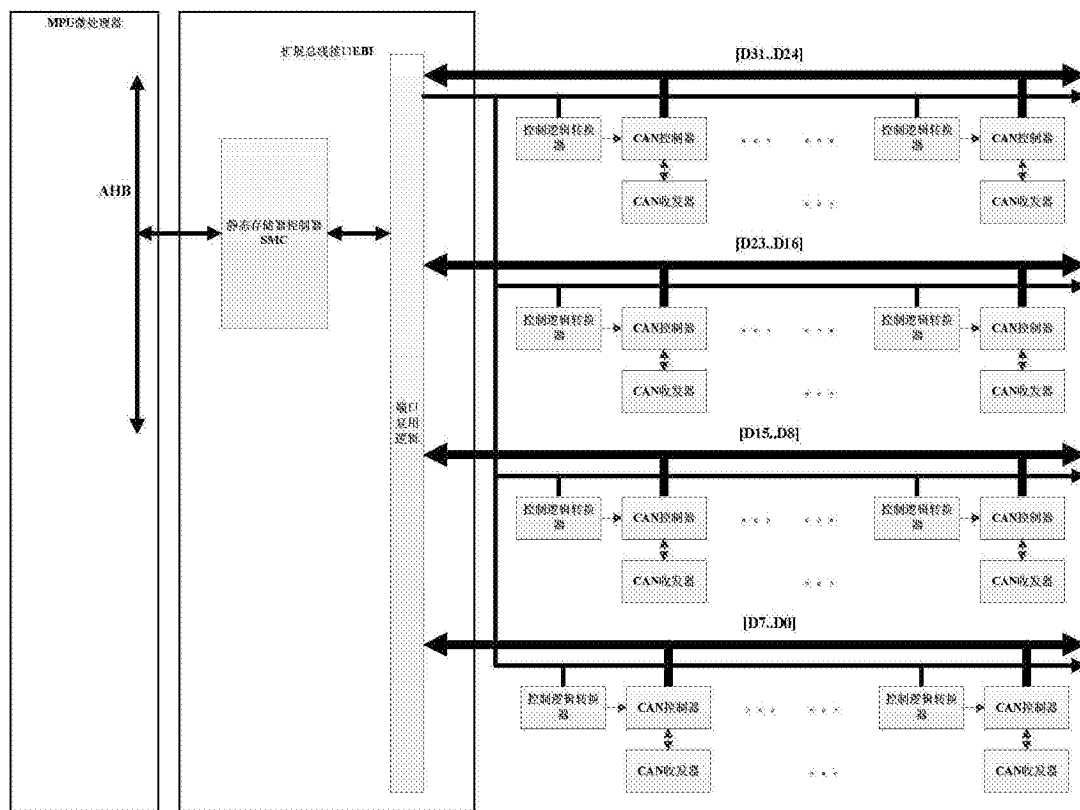


图4

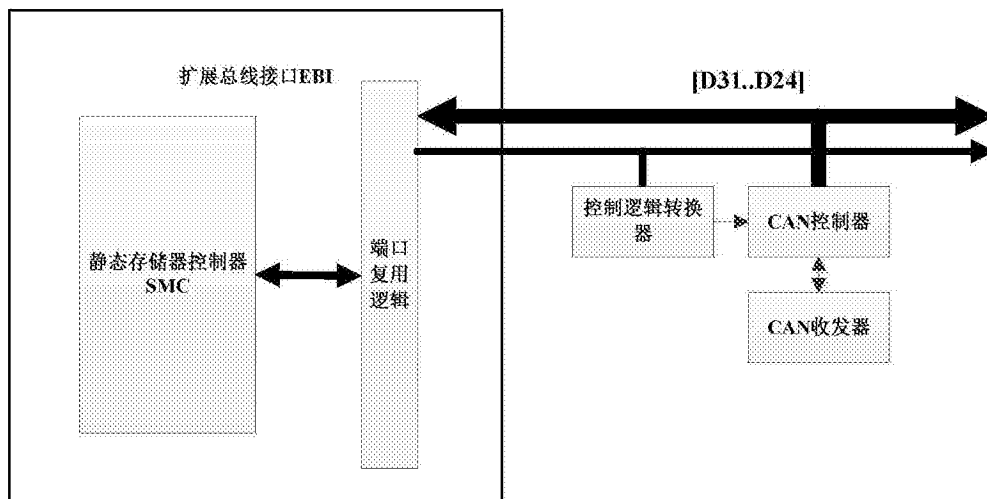


图5

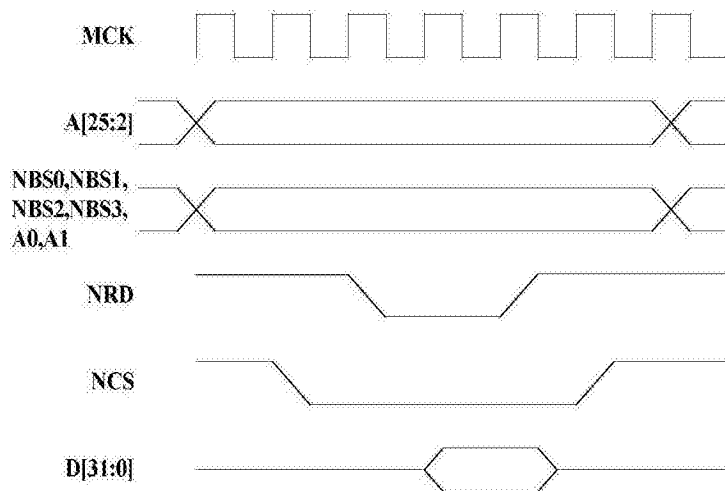


图6

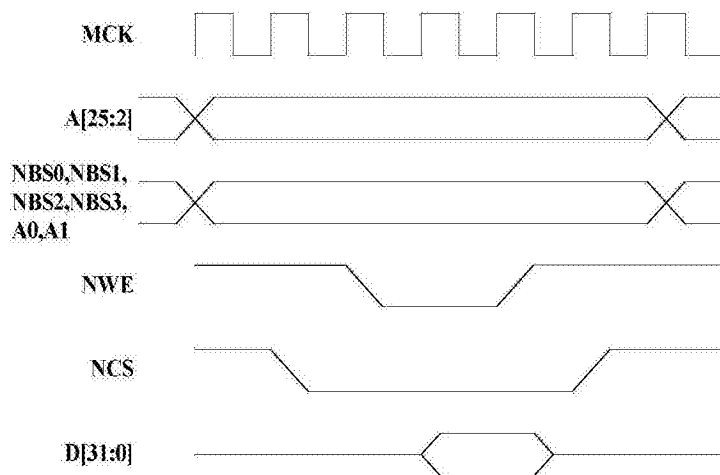


图7

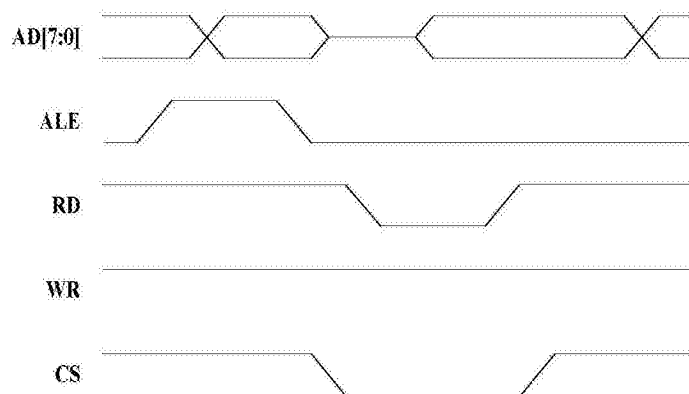


图8

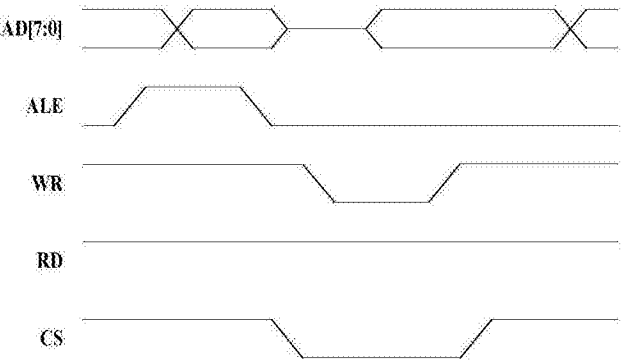


图9

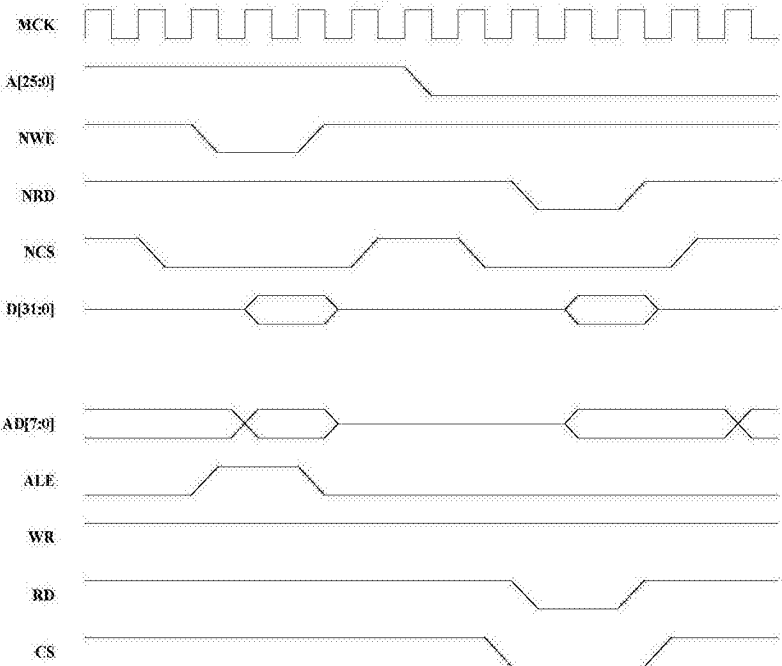


图10

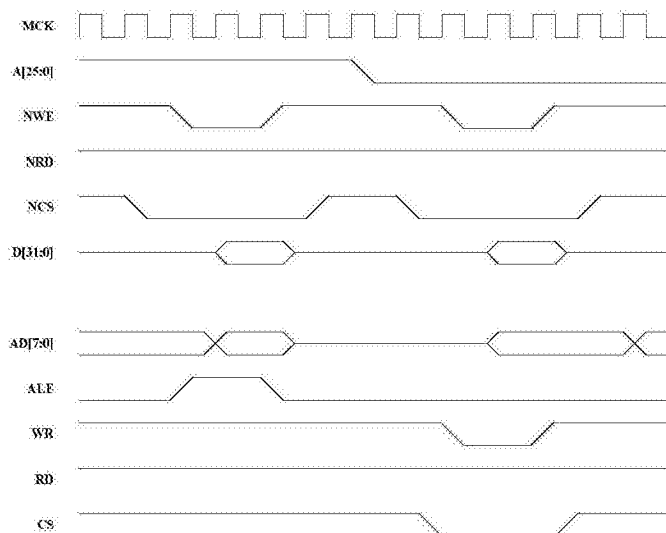


图11

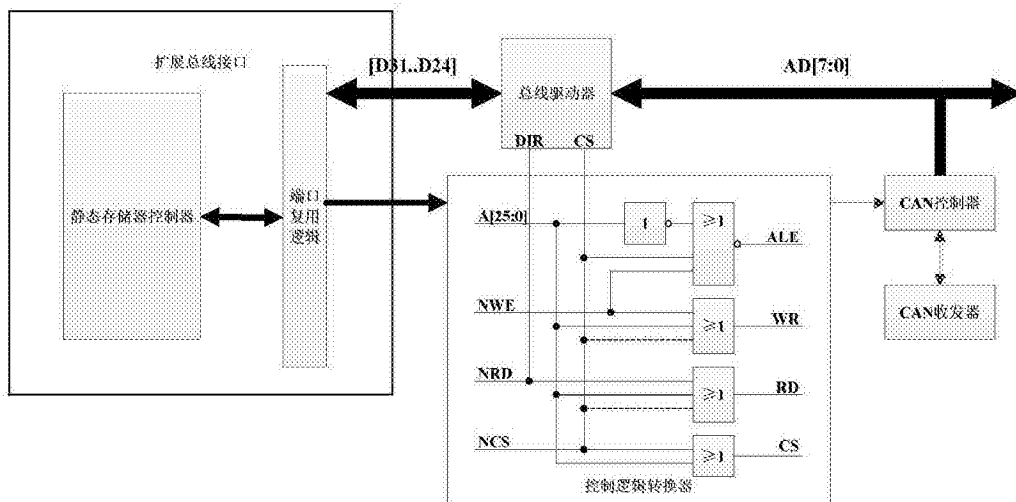


图12

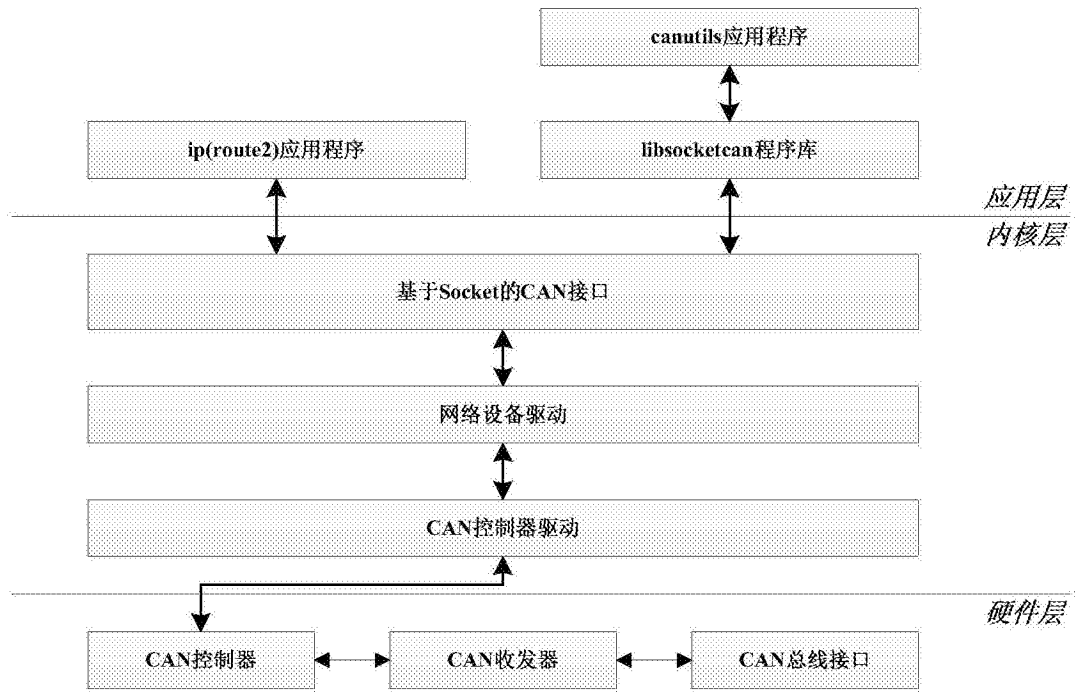


图13