(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 101052937 B
(45)授权公告日 2011.06.08

(21)申请号 200580037695.X
(22)申请日 2005.08.15
(30)优先权数据
10/936,142 2004.09.08 US
(85)PCT申请进入国家阶段日
2007.04.30
(86)PCT申请的申请数据
PCT/US2005/029198 2005.08.15
(87)PCT申请的公布数据
WO2006/028670 EN 2006.03.16
(73)专利权人 费希尔-罗斯蒙德系统公司
地址 美国得克萨斯州
(72)发明人 布赖恩·A·弗朗查克
罗杰·R·本森
(74)专利代理机构 中科专利商标代理有限责任公司 11021
代理人 王波波
(51)Int.Cl.
G06F 3/00(2006.01)
(56)对比文件
US 6539024 B1,2003.03.25, 全文.
CN 1504892 A,2004.06.16, 全文.
CN 1351784 A,2002.05.29, 全文.
US 2003/0112818 A1,2003.06.19, 全文.
审查员 王骞

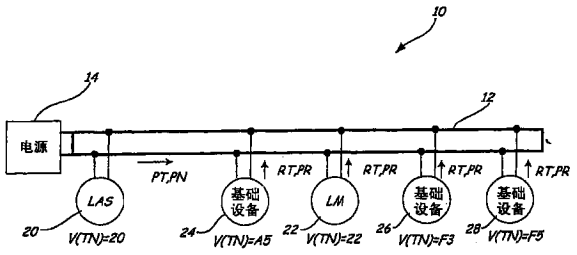
权利要求书 2 页 说明书 13 页 附图 5 页

(54)发明名称

从数据队列中丢弃部分接收的消息

(57)摘要

在进程控制系统中的设备在通信介质分段上对数据消息进行通信。每个设备包括通信控制器，通信控制器包括接收数据存储器和多个接收消息对象。当接收到消息时，激活接收消息对象并且将接收数据存储器中的写入指针的初始位置存储在接收消息对象中。在接收消息期间，如果硬件设备确定要忽略所述消息，则使接收消息对象无效，因此防止再将消息写入接收数据存储器。然后优选地使写入指针返回至存储在接收消息对象中的它的初始位置。



CN 101052937 B

1. 一种用于在通信介质上通信的设备，所述设备包括：

介质连接单元 MAU，用于在通信介质上接收和发送消息；

中央处理单元 CPU，用于处理接收消息中所包含的数据和创建将包含在发送消息中的数据；和

通信控制器，用于连接在 MAU 和 CPU 之间，所述通信控制器包括数据队列和接收消息对象的队列，所述数据队列用于存储在通信介质上接收的多个消息，每个接收消息对象顺序与数据队列中存储的消息相对应；

其中，当在通信介质上接收到消息时，激活第一接收消息对象以允许将消息存储在数据队列中，在激活第一接收消息对象时，将写入指针的初始位置存储到当前位置指针中；

其中，在消息接收期间，如果要丢弃消息，则使第一接收消息对象无效以防止再将消息存储到数据队列中；以及

其中，如果要丢弃消息，则将数据队列的写入指针恢复为当前位置指针所存储的初始位置。

2. 如权利要求 1 所述的设备，其中，设定有效标记以激活第一接收消息对象，清除有效标记以使第一接收消息对象无效。

3. 如权利要求 1 所述的设备，其中当接收到消息的有效结束信号时，激活第二接收消息对象。

4. 如权利要求 1 所述的设备，其中数据队列存储在环形缓冲器中。

5. 一种在接收整个消息之前对通信设备中的消息进行过滤的方法，该方法包括：

在接收消息之前启用第一接收消息对象；

将接收数据存储器中的写入指针的初始位置存储在当前位置指针中；

从写入指针的初始位置开始，将消息写入接收数据存储器；

当将消息写入接收数据存储器时，写入指针的位置递增；

在完成接收数据存储器中的消息接收之前，当硬件设备确定将丢弃所述消息时，禁用第一接收消息对象以防止再将所述消息存储在接收数据存储器中；以及

在完成消息的接收之前，如果禁用第一接收消息对象，则使写入指针返回当前位置指针所存储的初始位置。

6. 如权利要求 5 所述的方法，其中，在从消息接收到有效开始信号时，启用所述第一接收消息对象。

7. 如权利要求 5 所述的方法，还包括：

通过递增读取指针来读取存储到接收数据存储器的消息；以及

在完成消息的接收之前，如果禁用第一接收消息对象，则使写入指针返回当前位置指针所存储的初始位置，之后，将读取指针的值设置为等于写入指针的值，以确保在丢弃部分接收到的消息之后读取指针和写入指针处于相同位置。

8. 如权利要求 5 所述的方法，还包括：

当接收到消息的有效结束信号时，启用第二接收消息对象。

9. 如权利要求 5 所述的方法，其中启用第一接收消息对象包括：设定第一接收消息对象的有效标记。

10. 如权利要求 5 所述的方法，其中禁用第一接收消息对象包括：清除第一接收消息对

象的有效标记。

11. 一种从通信设备的数据队列中清除接收消息的方法，所述数据队列用于存储在通信介质上接收的多个消息，所述方法包括：

在接收消息之前，激活第一接收消息对象；

将数据队列的写入指针的初始位置存储到当前位置指针中；

从写入指针的初始位置开始，将接收消息写入数据队列；

当向数据队列写入接收消息时，写入指针的位置递增；和

在数据队列接收消息完成之前，确定是否要丢弃消息；

如果要丢弃消息，则使第一接收消息对象无效以防止再将消息存储在数据队列中；

在完成消息的接收之前，如果使第一接收消息对象无效，则使写入指针返回至当前位置指针所存储的初始位置，以便由通信设备接收的下一消息盖写所述接收消息。

从数据队列中丢弃部分接收的消息

技术领域

[0001] 本发明涉及用于进程控制系统的现场仪表和其它设备中的通信控制器。特别地，本发明是一种用于从接收数据存储器中丢弃部分接收的消息的系统和方法。

背景技术

[0002] 在典型的工业工厂中，采用分布式控制系统 (DCS) 来控制在工厂执行的许多工业进程。典型地，该工厂具有集中式控制室，该控制室具有带有如计算领域中已知的用户 (I/O)、盘 I/O 和其它外设的计算机系统。与计算系统相连的是控制器和进程 I/O 子系统。

[0003] 所述进程 I/O 子系统包括多个 I/O 端口，I/O 端口连接至遍及工厂的各种现场设备。控制领域公知的现场设备包括各种类型的分析仪器、硅压力传感器、电容性压力传感器、电阻性温度探测器、热电偶、应变计、限位开关、通 / 断开关、流量变送器、压力变送器、电容电平开关、磅秤、传感器、阀门定位器、阀门控制器、执行器、螺线管和指示灯。如这里所使用的，该术语“现场设备”涵盖了这些设备以及在分布式控制系统中执行功能的控制领域公知的任何其它设备。

[0004] 传统上，模拟现场设备通过二线双绞线电流环路连接至控制室，且各设备由单独的二线双绞线连接至控制室。模拟现场设备能够在指定范围内响应或发送电信号。在典型的结构中，通常在双绞线的二线之间具有大约 20-25 伏特的电压差，而流经环路的电流为 4-20 毫安。将信号发送至控制室的模拟现场设备调节流经电流环路的电流，而且该电流与所感应的进程变量成比例。另一方面，在控制室的控制下执行操作的模拟现场设备由经过环路的电流的强度所控制，该电流强度由受控制器所控制的进程 I/O 端口调节。传统的具有有源电子器件的二线模拟设备也可以从环路接收高达 40 毫瓦的功率。典型地，使用四线将需要更高功率 的模拟现场设备连接至控制室，四线中的二线向设备提供能量。正如二线设备一样，作为四线设备的这类设备在本领域中是已知的而且功率不受限制。

[0005] 相反，传统的分散现场设备发送或响应二进制信号。典型地，分立的现场设备以 24 伏特信号（或直流或交流）、110 或 240 伏特交流信号或 5 伏特直流信号运行。当然，可以设计分立设备，使其根据特殊控制环境所需的任何电力规范而运行。分立输入现场设备仅仅是一个开关，其接通或中断与控制室的连接，而分立输出现场设备将根据来自控制室的信号的存在或不存在而进行操作。

[0006] 从历史上看，大多数传统的现场设备已具有直接与现场设备所执行的主要功能相关联的信号输入或信号输出。例如，由传统的模拟电阻温度传感器所实现的唯一功能是通过调节流经二线双绞线的电流来传送温度，而由传统的模拟阀门定位器所实现的唯一功能是基于流经二线双绞线的电流强度在打开和关闭位置之间定位阀门。

[0007] 最近，将数字数据添加到电流环路上的混合系统已经用在分布式控制系统中。控制技术领域中已知的一种混合系统是高速可寻址远程传感器 (HART)，其与 Bell 202 调制解调器的规范类似。该 HART 系统使用电流环路中的电流强度来感测进程变量（如在传统系统中一样），并且还将数字载波信号添加到电流环路信号上。所述载波信号相对较慢，并

且可以提供每秒大约 2-3 次更新的速率的二级进程变量的更新。通常，数字载波信号用于发送二级和诊断信息，而不用于实现现场设备的主控功能。载波信号上所提供的信息包括例如二级进程变量、诊断信息（包括传感器诊断、设备诊断、接线诊断和进程诊断）、运行温度、传感器温度、校准信息、设备 ID 号、结构材料、配置或编程信息等等。因此，单个混合现场设备可以具有多个输入和输出变量，并且可以实现多种功能。

[0008] HART 是产业标准的非专有系统。然而，它相对较慢。产业中其它企业已经开发了更快的专有数字发送方案，但是通常竞争者不能使用或无法获得这些方案。

[0009] 最近，由美国仪表协会 (ISA) 规定了一种更新的控制协议。这种新协议一般被称为现场总线 (Fieldbus)。现场总线是一种多分支串行数 字双向通信协议，旨在将分布式控制系统中的现场仪器和例如监视和仿真单元的其它进程设备连接。现场总线实现了比之前的进程控制环路方法更强的数字通信，同时保持了向与现场总线环路相连的进程设备供电并同时满足固有的安全要求的能力。

[0010] 两种合理标准化的工业现场总线协议是基础现场总线 (FoundationFieldbus) 和现场总线 (Profibus)。现场总线协议的物理层由美国标准 ISA-S50.02-1992 的仪表协会规定，其草案 2 扩展发表于 1995 年。该现场总线协议规定了两个子协议。H1 现场总线网络以高达每秒 31.25 千比特 (Kbps) 的速率发送数据并向与该网络相连的现场设备供电。H1 物理层子协议由 1992 年 9 月批准的 ISA 标准第二部分的 11 条所规定。H2 现场总线网络以高达每秒 2.5 兆比特 (Mbps) 的速率发送数据，但是不向与该网络相连的现场设备供电，而且具有冗余的传输介质。

[0011] 现场总线具有用于对海量进程数据进行数字通信的显著性能。因此，不断地需要开发能够将现场总线通信效力最大化的进程控制设备。

发明内容

[0012] 本发明是一种系统和方法，用于从能够存储多个消息的数据队列中丢弃部分接收的消息。当接收消息时，将写入指针的初始位置存储在数据队列中。然后从写入指针的初始位置开始，将所述接收消息写入数据队列。当将接收消息写入数据队列时，写入指针的位置递增。如果要丢弃该消息，则使写入指针返回至初始位置，以便由通信设备接收的下一消息将盖写接收消息。

[0013] 在优选实施例中，数据队列被存储在通信控制器的接收数据存储器中。所述通信控制器还包括多个接收消息对象。当接收消息时，激活接收消息对象，并且将接收数据存储器中的写入指针的初始位置存储在有效接收消息对象中。然后，从写入指针的初始位置处开始，将接收消息写入接收数据存储器。当将消息写入接收数据存储器时，递增写入指针位置。如果硬件设备确定将忽略消息，则禁用有效接收消息对象，因此防止再将消息写入接收数据存储器。然后优选地，写入指针返回至存储在接收消息对象中的其初始位置。

附图说明

[0014] 图 1 是通信介质分段上设备之间的数字通信的进程控制系统的示意图。

[0015] 图 2 示出了图 1 的进程控制系统的设备之间的通信的消息格式。

[0016] 图 3 是进程控制系统的设备的方框图。

[0017] 图 4 是图 3 的设备的通信控制器的功能方框图。

[0018] 图 5 是用于处理在通信介质分段上所接收的数据分组的接收 / 发送事件管理器的功能方框图。

具体实施方式

[0019] 进程控制系统概况

[0020] 本发明涉及用于进程控制系统的现场装置和其它设备中的通信控制器。所述通信控制器的目的是执行消息链路层处理和定时器管理的实际部分，并因此释放应用处理器或 CPU 以执行其它功能。为了进行详细描述，将在使用基础现场总线通信协议的系统的背景下描述所述通信控制器，虽然它具有对于基于分组的通信协议的一般适用性。

[0021] 所述现场总线物理层定义了以物理层协议数据单元 (PhPDU) 的格式对通信协议数据进行发送和接收的物理装置的电子特性。此外，现场总线物理层规定符号编码、消息帧化和错误检测方法。ISA 现场总线标准定义了三种信号收发的速度和两种连接模式。为了对此进行描述，将在 ISA S50.02 标准第 2 部第 11 条中所定义的 H1 物理层的背景下描述本发明。该条涵盖 31.25Kbps、电压模式、有线介质和低功率选项。该选项使得连接至通信介质的设备能够从通信介质接收它的操作功率。所述物理层能够符合对于危险环境的内部安全要求。所述协议根据由标准定义的电压和电路限制，在低等级双绞线上运行并支持多个设备。

[0022] 图 1 示出了一种典型的进程控制系统 10，包括分段 12、电源 14 和五个设备 ：链路活动调度器 (LAS) 设备 20、链路主设备 (LM)22 以及基础设备 24、26 和 28。分段 12 可以在单独一对线路上支持多达 32 个设备。典型地，基于环路执行速度、功率和内部安全要求，分段 12 将具有 4-16 个设备。

[0023] LAS 设备 20 为分段 12 上的设备之间的所有通信保持中央调度。通过将强制数据 (CD) 数据链路协议数据单元 (DLPDU) 发送至每个设备以发送后周期数据，然后调度每个设备发送后周期数据，LAS 设备 20 改进了整体通信的可靠性。LAS 设备 20 用作分段 12 上的数据链路时间 (DL-time) 的本地来源。DLPDU 是通过分段 12 传送的 PhPDU 消息的数据内容。

[0024] LM 设备 22 被配置成如果 LAS 设备 20 发生故障或变得不可操作，则承担 LAS 设备 20 的责任。虽然图 1 只示出了 LM 设备 22，但是在分段上可存在两个以上链路控制设备。这使得下列情形成为可能，如果链路活动调度器和第一链路主设备二者均发生故障，则第二链路主设备可以承担链路活动调度器的责任。在链路活动调度器发生故障之后，链路主设备承担链路活动调度器的功能。

[0025] 每个设备都具有被称为 V(TN) 的唯一地址，它代表本地节点 ID(此节点)。在图 1 所示的例子中，LAS 设备 20 具有地址 V(TN) ＝ 20 ；LM 设备 22 具有地址 V(TN) ＝ 22 ；基础设备 24 具有地址 V(TN) ＝ A5 ；基础设备 26 具有地址 V(TN) ＝ F3 ；以及基础设备 28 具有地址 V(TN) ＝ F5。

[0026] LAS 设备 20 将传递令牌 (PT) 和探测节点 (PN) 消息发送至分段 12 上的所有设备。其它设备 (LAS 设备 22 和基础设备 24、26、28) 中的每一个适当地将返回令牌 (RT) 和探测响应 (PR) 消息发回至 LAS 设备 20。

[0027] 每个基础设备 24、26、28 只需要查看其自己的由 LAS 设备 20 发送的 PT 和 PN 消息。PT 和 PN 消息具有编码在 DLPDU 的第二字节中的指定地址 (DA)。LAS 设备 20 向分段 12 上的所有设备一次发送一个记号 (PT) 或探测一个节点 (PN)。

[0028] 如果在基础设备 24、26 或 28 接收到带有与设备的唯一地址相等的指定地址 (DA = V(TN)) 的 PT 消息，则它将向 LAS 设备 20 反馈 RT 消息。如果基础设备 24、26 或 28 接收具有 DA = V(TN) 的 PN DLPDU，则需要它反馈 PR 消息。

[0029] 从 LAS 20 发送 PT 和 PN 消息以及向 LAS 20 发送 RT 和 PR 消息在分段 12 上创建了若干消息，特定的基础设备 24、26、28 并不需要接收这些消息并对其采取行动。每个基础设备 24、26、28 只需要响应于寻址到该特定设备的 PT 和 PN 消息。持续地受到来自 LAS 20 的寻址到其它设备的 PT 和 PN 消息以及来自其它设备的寻址到 LAS 设备 20 的 RT 和 PR 消息的干扰，会产生过度的处理时间来解决这些“多余中断”。利用基础设备 24、26 和 28，DLPDU 过滤可以用于减少基础设备必须处理的中断的数量。另一方面，LAS 设备 20 必须处理分段 12 上的每个消息。

[0030] 分段 12 上的所有设备将数据作为曼彻斯特编码基带信号发送到分段 12 上。由于曼彻斯特编码，“0”和“1”分别表示比特周期中间发生的从低到高和从高到低的转换。对于现场总线，标称比特时间是 32 微秒 (μsec)，转换以 16μsec 的频率发生。曼彻斯特编码规则已经被扩大至包括两种附加符号，即非数据加 (N+) 和非数据减 (N-)，其中在比特周期过程中不发生转换，而且曼彻斯特编码基带信号保持高 (N+) 或低 (N-)。

[0031] 消息格式

[0032] 图 2 示出了用于在分段 12 上发送消息的物理层协议单元 (PhPDU) 的格式。PhPDU 包括前同步、开始分隔符 (SD)、数据链路协议数据单元 (DLPDU) 和结束分隔符 (ED)。所述前同步是 PhPDU 消息的前面若干比特。现场总线规范允许 1-8 个字节的前同步。接收消息的设备使用前同步，以与输入的消息同步。如图 2 所示，前同步的首字节序列是 10101010。

[0033] 开始分隔符 (SD) 紧接着前同步。这里每个消息有一个 SD。现场总线规范要求 SD 具有非字符数据 (N+ 和 N-)，其总是以互补对的方式出现在 SD 消息中。此编码方案使得 SD 唯一并且不可能与消息的数据部分 (DLPDU) 混淆。图 2 中所示 SD 的序列为 1N+N-10N-N+0。

[0034] DLPDU 是长度可变的消息。它以帧控制 (FC) 字节作为第一字节以及以帧校验序列 (FCS) 校验和作为其最后两个字节。DLPDU 的长度可变，最小是三个字节 (在 RT 消息的情况下)，大到例如大约 300 字节的超时传输 (jabber) 限制。

[0035] 结束分隔符 (ED) 在 DLPDU 之后。它表示在分段 12 上的发送 PhPDU 消息的最后字节。类似于 SD，ED 包括呈互补对的非字符数据。这个编码方案使得 ED 唯一并且不可能与 DLPDU 混淆。图 2 中所示结束分隔符的序列为 1N+N-N+N-101。

[0036] 图 2 还示出了载波检测信号。载波检测信号的目的是指示 :何时 (a) 分段 12 上存在输入的 PhPDU 消息，或 (b) 设备正在将消息发送到分段 12 上。

[0037] 发送开始 (SOT) 发生在这样的时刻 :发送使能 (TxE) 变得有效，即当 PhPDU 的前同步被首先提供给分段 12 时。

[0038] 活动开始 (SOA) 发生在载波检测信号变得有效而且已经稳定了至少一个比特时间或两个比特时间 (大约 16-31μsec) 之后。这个时间取决于载波检测相对于接收消息的设备的内部时钟何时变得有效。这使得设备的通信控制器可以忽略在前同步的前端十分容

易发生的噪音干扰。附加时间用于与比特边界同步，以消除分段 12 上的短噪声脉冲被误译为有效的可能。对于发送消息，在发送使能变得有效之后发生 SOA（即 PhPDU 的前同步被提供给分段 12）。

[0039] 消息开始 (SOM) 发生在针对接收消息检测到 FC 字节时的第一比特的开始处。

[0040] SOM xmt 是消息发送开始，其发生在针对发送消息检测到 FC 字节时的第一比特的开始处。

[0041] SOMf 是所接收已过滤的 DLPDU 的 SOM。这发生在当设备中的通信控制器已经检测到足够的信息以确定过滤输入消息的时候。

[0042] 消息结束 (EOM) 发生在接收消息中遇到的 ED 的最后比特的结尾。发送结束 (EOT) 发生在发送消息的 ED 的最后比特的结尾。

[0043] 有效结束 (EOA) 发生在载波检测已经变得无效的时候。发送 DLPDU 和接收 DLPDU 二者均发生 EOA。

[0044] 设备结构

[0045] 图 3 示出了基础设备 24 的通信部分的方框图，其表示每一个设备 20-28 中的结构。基础设备 24 包括中央处理器 (CPU)30、随机存取存储器 (RAM)32、闪存 34、通信控制器 36 和媒体连接单元 (MAU)38。

[0046] 在图 3 所示的实施例中，CPU 30 是微处理器，例如摩托罗拉 68LC302、摩托罗拉 Mcore 2075、摩托罗拉 PowerPC 850、Atmel Thumb 处理器 AT91M40800 和其它。CPU30 是 8 比特或更高的处理器。

[0047] 在图 3 所示的实施例中，通信控制器 36 是特殊应用集成电路 (ASIC)，其作为 MAU 38 和 CPU 30 之间的接口工作。它向连接至现场总线分段 12 的外部模拟电路发送和从连接至现场总线分段 12 的外部模拟电路接收已编程的曼彻斯特数据。在从 MAU 38 接收到串行数据之后，通信控制器 36 对数据解码，将数据形成为字节，去掉前同步、SD 和 ED（和可选地 FCS 字节），并为链路层提供消息数据以供读取。对于数据发送，通信控制器 36 从链路层接收 DLPDU 数据的字节，并添加前同步、SD，可选地产生 FCS，以及添加 ED。接着通信控制器 36 串行形成已编码的曼彻斯特数据，将所述数据发送至 MAU 38 以供在现场总线分段 12 上发送。

[0048] 通过下列四种信号提供通信控制器 36 和 MAU 38 之间的通信 ：RxS、RxA、TxS、TxE。RxS 是所接收的曼彻斯特编码串行数据。RxA 是接收数据的载波检测信号。TxS 是发送的已编码串行数据。TxE 是发送使能信号。

[0049] 在本发明的其它实施例中，可以在具有 CPU 30 的公共集成电路上形成通信控制器 36。而且，在有些实施例中，RAM 32 和闪存 34 也可以和 CPU 30 合并。在 LAS 设备 20 的情况下，CPU 30、RAM 32 和闪存 34 可以是进程控制系统 10 的主计算系统的一部分。

[0050] MAU 38 向现场总线分段 12 提供网络连接。MAU 38 可以是集成电路，或者是可用于形成 MAU 38 的分立组件。

[0051] 通信控制器 36

[0052] 图 4 是通信控制器 36 的功能方框图。在此实施例中，通信控制器 36 包括反跳电路 42、数字锁相环 (PLL)44、前端状态机 46、接收消息过滤 48、接收先入先出 (FIFO) 存储器 50、发送状态机 52、发送 FIFO 存储器 54、发送驱动器电路 56、接收 / 发送事件管理器 58、寄

存器 60、时钟生成电路 62、振荡器 64、定时器 68 和 CPU 接口电路 70。

[0053] 当 MAU 38 检测到输入消息时，在 RxA 输入处向通信控制器 36 提供载波检测信号，而且在 RxS 输入处提供输入的异步曼彻斯特数据。将 RxA 和 RxS 输入提供给前端状态机 46。数字 PLL 44 从输入的串行曼彻斯特编码数据中再生和重建时钟。然后将此重建的时钟用于对前端状态机 46 计时。

[0054] 前端状态机 46 检测输入的串行比特流 RxS。它去掉前同步、SD 和 ED，并将 DLPDU 存储至接收 FIFO 存储器 50 中。前端状态机 46 和接收消息过滤 48 一起可以被配置用于滤除特殊帧控制，并加上寻址到其它设备的探测节点 (PN) 及传递令牌 (PT)。前端状态机 46 跟踪已经被写入接收 FIFO 存储器 50 中的字节数量。在每个消息的结尾处自动验证 FCS，并且可选地可以将 FCS 存储至接收 FIFO 存储器 50 中。

[0055] 前端状态机 46 还提供表示它已经检测的特殊事件的信号。这些包括 SOM、SOMf、EOM、SOA 和 EOA 事件脉冲。

[0056] 当 RxA 链路变得有效时前端状态机 46 被激活。然后前端状态机 46 与前同步字段的沿同步，并将 RxS 信号的曼彻斯特编码数据解码。SOA 事件指示前端状态机 46 已经启动。

[0057] 在检测到前同步之后，前端状态机 46 等待开始分隔符 (SD) 序列。在已经检测到 SD 之后，前端状态机 46 将串行数据流转换为八位组，并以 8 比特字节将其写入接收 FIFO 存储器 50。前端状态机 46 继续将数据的新八位组写入接收 FIFO 存储器 50 中，直到检测到结束分隔符 (ED)，或直到接收 FIFO 存储器 50 满为止。

[0058] 当已经检测到 ED 时，前端状态机 46 等候 RxA 线路变得无效，而这是由 EOA 事件指示的。

[0059] 当 RxA 线路无效时，前端状态机 46 返回其初始状态。它在现场总线分段 12 上的下一次有效之前（即在 RxA 处再次提供载波检测信号之前）保持在此初始状态中。

[0060] 过滤电路用于基础设备，以减少加载在对于设备不重要的消息上的 IRQ。相反地，配置作为 LAS 的设备必须接收分段上的所有消息，并因此必然过滤被禁止。当过滤被禁止时，所有接收到的消息将被存储在接收 FIFO 存储器 50 中，并将被传递至寄存器 60 然后至 CPU 30。SOMf 是对于接收到的已过滤 DLPDU 的消息开始信号。它发生在前端状态机 46 已经确定已经对接收消息检测到足够的信息以确定输入的消息将被过滤的时候。

[0061] 由于可以过滤，则不将被过滤的消息存储在接收 FIFO 存储器 50 中。对于已过滤的消息，将不产生 SOMf，因此将不发生事件或 IRQ。

[0062] 已过滤的消息的例子有返回令牌 (RT)、空闲、请求间隔 (RI)，并且总是丢弃探测响应 (PR)DLPDU 消息。根据帧控制 (FC) 字节来识别这些消息。如果消息中的目标地址与设备地址匹配，则接收传递令牌 (PT) 和探测节点 (PN) 消息。如果消息中的目标地址与设备地址不匹配，则丢弃 PT 和 PN 消息。

[0063] 基于 FC 字节和根据目标地址来过滤消息字节的能力通过限制 CPU30 必须处理的中断请求 (IRQ) 的数量，减少了软件中断的加载。

[0064] 前端状态机 46 和接收 FIFO 存储器 50 用于解析来自 MAU 38 的串行数据帧。CPU 30 从接收 FIFO 存储器 50 读取数据，并将其放置在它的本地存储器空间以对接收 DLPDU 进行解码。

[0065] 接收 FIFO 存储器 50 具有 8 比特宽的 63 字节。接收 FIFO 存储器 50 将存储多达 3 个完整的接收消息的所有 DLPDU 字节（总体多达 63 字节）。前端状态机 46 将来自已过滤 RxS 信号的串联数据流解码，并将其转换为 8 比特并行格式字节。在形成该字节之后，前端状态机 46 创建写入脉冲，其将已编码数据存储在写入指针所指向的位置。在写入操作完成之后，使写入指针递增以存储下一个 DLPDU 字节。

[0066] CPU 30 通过读取指针与接收 FIFO 存储器 50 连接。任何来自寄存器 60 的接收 FIFO 寄存器（其包含实际的 DLPDU 数据）的读取将来自接收 FIFO 存储器 50 的 8 比特数据立即放置在供 CPU 30 读取的数据总线上。在读取操作完成之后，使读取指针递增。在接收 FIFO 存储器 50 为空之前可以继续上述操作。

[0067] 为了防止在接收 FIFO 存储器 50 中发生上溢情况，在寄存器 60 中有这样一个寄存器，其可以在接收 FIFO 存储器 50 接近满的情况下产生 IRQ。用于产生 IRQ 的阈值是可配置的。

[0068] 发送状态机 52 读取将从发送 FIFO 存储器 54 发送来的 DLPDU 数据。自动插入前同步、SD 和 ED。为了启动发送状态机 52，可选地需要激活 interPDU 事件触发器或下一调度事件触发器，以开始发送操作。发送状态机 52 跟踪已经发送的字节的数量。如果存在下溢或发送计数破坏 (violation)，则将指示错误状况。可选地，可以自动将 FCS 作为 DLPDU 的最后两个字节进行发送。

[0069] 发送状态机 52 对通过 TxS 线路上的接口电路 70 提供给 MAU 38 的曼彻斯特串行数据进行编码，以提供在现场总线分段 12 上。发送状态机 52 还在发送第一前同步的第一比特的时刻维护发送使能 (TxE) 线路，直到 ED 的最后字节出现。当发送状态机 52 维护 TxE 线路时，它还产生发送开始 (SOT) 事件信号，并在 TxE 线路返回至无效时产生发送结束 (EOT) 事件信号。

[0070] 发送 FIFO 存储器 54 将存储待发送消息所需要的所有 DLPDU 字节，总体多达 63 字节。可以设定可配置的阈值，以在发送 FIFO 存储器 54 几乎空的时候发送 IRQ 来通知 CPU 30。这样，如果需要发送多于 63 字节，则通知 CPU 30 以便其可以向发送 FIFO 存储器 54 添加更多的数据。在所有 DLPDU 字节被写入之前继续上述操作。CPU 30 使用写入指针来向发送 FIFO 存储器 54 写入，而发送状态机 52 使用读取指针从发送 FIFO 存储器 54 读取字节。

[0071] 通信控制器 36 作用于事件，并必须能够处理多个事件的发生。事件的例子包括 SOM、EOM、或接收消息的 EOA 或发送消息的 EOT。接收 / 发送事件管理器 58 管理总体上多达 3 个接收消息和一个发送消息所发生的所有事件。

[0072] 如图 4 所示，接收 / 发送管理器 58 包括三个接收消息对象，标为 rcvmsg1、rcvmsg2 和 rcvmsg3，以及一个发送消息对象，标为 xmtmsg。此外，接收 / 发送管理器 58 包括消息队列管理器 (MsgQmngr)80、事件管理器 (EventMngr)82、发送管理器 (xmtmngr)84 和事件 MUX 86。

[0073] 接收 FIFO 存储器 50 能够存储多达三个完整的接收消息的 DLPDU 字节。这三个消息的每一个具有相应的对象，即 rcvmsg1、rcvmsg2 和 rcvmsg3。每个对象包含其相应接收消息发生的所有 IRQ 的状态、消息错误和时间记录。这些信息构成该消息的事件数据。

[0074] 将发送消息发生的所有 IRQ 的状态、消息错误和时间记录存储在 xmtmsg 对象中。所存储的信息构成发送消息的事件数据。

[0075] MsgQmngr 80 控制三个接收消息的选择和使能。一次只能有一个 rcvmsg 对象是有效的。MsgQmngr 80 使事件与有效的接收消息相关联。 在 CPU 30 已确认其它三个消息之前接收到第四个消息的情况下，MsgQmngr 80 在读取或确认事件数据之前，禁止接收其它任何消息。EventMngr 82 管理事件发生的顺序。随着事件发生，EventMngr 82 给每个事件分配一个发生顺序标识 (OOO ID)。这使得 CPU 30 能够按照事件发生的顺序一次读取一个事件。CPU 30 必须在每个事件发生时确认每个事件。在已经确认第一事件之后，接下来的事件将就绪以供 CPU 30 读取。

[0076] Xmtmngr 84 监视 interPDU 触发器 (InterPDU_trig) 和下一调度事件触发器，并将发送触发器命令 (Xmt_Trig_Cmd) 引入到发送状态机 52 以开始发送下一个消息。

[0077] 通信控制器 36 包括寄存器 60。这些标为 REG00-REG3F 的寄存器可以由 CPU 30 读取和写入。通过寄存器 60 还可以操纵中断 (IRQ)。

[0078] 时钟生成电路 62 接收外部时钟，并且或者使用该时钟或者使用来自其内部振荡器 64 的时钟，来生成通信控制器 36 所需的所有时钟信号。

[0079] 时钟生成电路 62 优选地具有及时地调整其节点定时器和其八位组定时器时钟速率的能力。这使得通信控制器能够使其节点时间与链路地址调度器 (LAS 20) 同步。八位组时间用于内部消息的定时，而节点时间用于在现场总线分段 12 上共享公共的时间感。

[0080] 定时器 68 分为两组，代表不同的时间感。被称为分段定时器的第一组定时器在来自 CPU 30 的软件控制下，基于由时钟生成电路 62 产生的可变的时钟速率而运行。被称为消息定时器的第二组定时器以固定速率时钟运行。

[0081] 在通信控制器 36 中存在两个分段定时器。第一分段定时器是节点定时器，其具有 31.25μsec(32kHz) 的时钟报时速率。节点定时器用于实现下一功能块执行时间、链路调度时间 V(LST) 和数据链路时间 (DL-Time)。

[0082] 第二分段定时器是八位组定时器，其具有 2μsec(500kHz) 的时钟报时速率。八位组定时器用于下一调度事件触发器（其与发送状态机 52 连接以便在特定时间发送消息）。当调整时钟速率时，节点和八位组定时器将以相同的速率相互跟踪。这是因为驱动节点定时器和八位组定时器的时钟信号是由公共可变时钟得到的。

[0083] 基于现场总线消息事件（发送和接收）而开始和停止消息定时器。消息定时器包括无效定时器、interPDU 延迟定时器、接收应答定时器、发送应答定时器和代表令牌 (delegated token) 恢复定时器。

[0084] 无效定时器是递减计数器。它用于测量两个 PhPDU 之间的空闲时间。无效定时器作用于已过滤的和未过滤的接收消息以及现场总线分段 12 上的任意发送消息。当被命令开始时，无效定时器每 16μsec 递减一次。无效定时器的开始点是根据载入寄存器 60 之一的可配置预加载设置点确定的。通过与接收或发送消息相关的事件可以取消或停止无效定时器的递减。如果定时器达到 0 或者期满，将生成 IRQ。在确认 IRQ 之前无效定时器将保持为 0。如果 IRQ 保持为高，则在确认 IRQ 之前将不会发生影响无效定时器的附加消息事件。

[0085] interPDU 延迟定时器是递增计数器。它用于与 V(MID) 阈值寄存器结合以实现现场总线 V(MID) 最小—interPDU 延迟，其确保在发送或接收消息之间的无传输的最小持续时间（或间隙时间）。interPDU 定时器受到现场总线分段上的已过滤的和未过滤的接收消息、以及任意发送消息的作用。当不存在现场总线活动时，interPDU 定时器将持续递增。如

果计数值等于或超过寄存器 60 中所存储的预定值，则 interPDU_trig 信号将变得有效。该信号用于确定已经满足了 interPDU 延迟时间。信号与 xmtmngr 84 相连以给出发送 DLPDU 可以开始的命令。

[0086] 接收应答定时器是递减计数器。它用于订阅设备以监视对强制数据 (CD)DLPDU 的及时响应。它还由设备用于在设备进入链路上时监视其自己的地址。当被命令开始时，接收应答定时器每 16μsec 递减一次。接收应答定时器的开始点是根据载入寄存器 60 之一的可配置预加载 16 比特设置点确定的。通过 SOM 或 SOT 事件，可以取消或停止接收应答定时器的递减。如果接收应答定时器达到 0 或者期满，将生成 IRQ。接收应答定时器需要启用 IRQ 以生成 IRQ。在确认 IRQ 之前接收应答定时器将保持为 0。如果 IRQ 保持为高，则在确认 IRQ 之前将不会发生会影响接收应答定时器的附加消息事件。

[0087] 发送应答定时器是递减计数器。它使得设备能够在发送若干 DLPDU（例如强制数据、传递令牌）之一后监视及时响应。当被命令开始时，发送应答定时器每 16μsec 递减一次。发送应答定时器的开始点是根据载 入寄存器 60 之一的可配置预加载设置点确定的。通过除了探测节点 (PN) 外的任何发送 DLPDU 的 SOM 事件或 SOT 事件，可以取消或停止发送应答定时器的递减。如果发送应答定时器达到 0 或者期满，将生成 IRQ。在确认 IRQ 之前发送应答定时器将保持为 0。如果 IRQ 保持为高，则在确认 IRQ 之前将不会发生会影响发送应答定时器的附加消息事件。

[0088] 代表令牌恢复定时器是递减计数器。它用于监视从另一个设备接收代表令牌的空闲时间。代表令牌恢复定时器作用于已过滤的和未过滤的接收消息以及现场总线分段 12 上的任意发送消息。当被命令开始时，代表令牌恢复定时器每 16μsec 递减一次。代表令牌恢复定时器的开始点是根据载入寄存器 60 之一的可配置预加载设置点确定的。通过与接收或发送消息相关的事件，可以取消或停止代表令牌恢复定时器的递减。如果代表令牌恢复定时器达到 0 或者期满，将生成 IRQ。在确认 IRQ 之前发送反馈定时器将保持为 0。如果 IRQ 保持为高，则在确认 IRQ 之前将不会发生会影响代表令牌恢复定时器的附加消息事件。

[0089] 丢弃部分接收的消息

[0090] 典型地，通信硬件负责对从网络接收的消息进行过滤，并且只将那些感兴趣的消息发送至软件。消息过滤对硬件造成一种特殊问题，即在决定是否过滤消息之前消息过滤器可能需要部分接收消息。如果消息过滤器决定过滤消息，则需要丢弃部分接收的消息。如果消息过滤器决定不过滤消息，则硬件需要将整个消息发送至软件，包括已经部分接收到的部分。

[0091] 图 5 是用于管理对由通信控制器 36 所接收的数据分组进行处理的接收 / 发送事件管理器 58 的功能方框图。接收 / 发送事件管理器 58 包括消息队列管理 (MsgQmngr)80、事件管理器 (EventMngr)82、事件 MUX 86、接收消息对象队列 100 和多路复用器 102、106 和 108。接收消息对象队列 100 包括用于三个标为 rcvmsg1、rcvmsg2 和 rcvmsg3 的接收消息对象的空间。接收 / 发送事件管理器 58 还包括当前位置指针 (CurPosition)116、指针使能逻辑 118、写入指针 120 和读取指针 122。图 5 中还示出接收 FIFO 存储器 50 和用于存储与接收和发送消息对象相关的事件数据的寄存器 60 的一部分 (Reg15-Reg1F)。

[0092] 接收 / 发送事件管理器 58 允许 CPU 30 依次读取与已经发生的各个消息相关联的

接收消息对象 (rcvmsg1、rcvmsg2 和 rcvmsg3)。每个接收消息对象包含消息属性的分类，包括消息 IRQ、错误状况和普通消息信息。与接收消息对象相关的 DLPDU 数据被存储在接收 FIFO 存储器 50 中。

[0093] 发送消息对象 (xmtmsg) 接受来自发送管理器 (xmtmngr)84、发送状态器 52、发送 FIFO 存储器 54 和定时器 68 的输入。从这些输入，CPU 30 可以读取在寄存器 60 的寄存器 Reg16、Reg17 和 Reg1B 中的发送 IRQ、错误状况和时间记录信息。为了简明，从图 5 中略去图 4 中的接收 / 发送事件管理器 58 示出的 xmtmngr 84、发送 FIFO 存储器 54 和发送状态器 52，因为它们只与发送消息相关。

[0094] MsgQmngr 80 允许特殊接收消息事件与来自分段 12 的有效接收消息相关联。这经由 MUX 102 进行控制，形成信号 RcvMsg1_sel、RcvMsg2_sel 和 RcvMsg3_sel。一次只有这些选择信号中的一个是有效的。因此，一次只有一个接收消息对象 (rcvmsg1、rcvmsg2 和 rcvmsg3) 具有与它相关的接收事件，同时消息是有效的。多个输入进入接收消息对象队列 100。这些信号包括读取指针 122 的当前位置值、写入指针 120、GetDataByte_in(用于读取 RcvFIFO) 和来自前端状态机 46 的信号、以及来自定时器 68 的当前时间值。这三个选择信号 (RcvMsg1_sel、RcvMsg2_sel 和 RcvMsg3_sel) 允许输入信号适当地只与一个接收消息对象相关联。

[0095] 如果在接收消息之后发生 EOA 事件脉冲同时在接收消息对象队列 100 中仍然存在 3 个未确认的接收消息对象，则 MsgQmngr 80 通过强制使 RcvMsgQMux_enb 信号为低 (无效) 来禁止接收第四个消息。还将 MsgQ_over 信号发送到指针使能逻辑 118。这防止第四个接收消息破坏接收 FIFO 存储器 50 中的 DLPDU 数据。为了再次启用 RcvMsgQMux_enb 信号，CPU 30 必须确认发生第一个接收消息对象 (即时间上最早发生的消息)。CPU 30 写入寄存器 60 中的寄存器 Reg15，产生 Event_Ack 信号 (事件确认)，并且随后根据 OR 门 110 的输出来产生 RcvMsg_Ack 信号。

[0096] EventMngr 82 控制事件 MUX 86 的选择，以在三个接收消息对象 (rcvmsg1、rcvmsg2 和 rcvmsg3) 和发送消息对象 (xmtmsg) 之一的读取中进行多路复用。CPU 30 可经由寄存器 60 的寄存器 Reg15-Reg1F，读取有效消息的接收消息对象。CPU 30 读取寄存器 Reg15 的两个较低比特，以确定下一个待处理的事件是否是接收消息 (“01”)、发送消息 (“10”) 或是否不存在要处理的事件 (“00”)。对于接收消息，MUX106 形成 6 个信号 : RcvMsg1_visible、RcvMsg2_visible、RcvMsg3_visible、RcvMsg1A_ckn、RcvMsg2_Ackn 和 RcvMsg3_Ackn。前面 3 个信号互相排斥，选择三个接收消息对象的一个以通过事件 MUX 86 传送以供 CPU30 处理。因此，一次只可以处理一个接收消息对象的相关事件信息和属性数据。以与从分段 12 接收的顺序相同的顺序来处理所述事件。

[0097] 信号 RcvMsg1_Ackn、RcvMsg2_Ackn 和 RcvMsg3_Ackn 用于在 CPU30 已经处理所有事件数据之后确认当前可视的接收消息对象。这三个信号被输入 OR 门 110，并且在 OR 门 110 的输出产生 RcvMsg_Ack 信号。RcvMsg_Ack 信号作为输入被提供给 MsgQmngr 80 和读取指针 122。

[0098] MsgQmngr 80 还控制 MUX 108，其用于创建信号 CurMsg_Ack(当前消息确认)。所述“当前消息”是分段 12 上的有效接收消息，CPU 30 正在处理该消息的事件数据 (即消息对于 CPU 30 可视)。CurMsg_Ack 由 MsgQmngr 80 使用，并 (通过指针使能逻辑 118) 控制

写入指针 120 和当前位置 (CurPosition)116。当接收消息对象是可视的并且接收消息对象的相应选择信号是有效的时（例如 RcvMsg1_visible ＝ RcvMsg1_sel ＝ 1),CurMsg_Ack 是有效的。

[0099] 经由GetDataByte_in信号,CPU30将读取指针122与接收FIFO存储器50连接。通过当前对于CPU 30可视以供读取的接收消息对象,CPU只可以访问读取指针122(即对于接收消息对象,RcvMsg_visible信号是有效的)。当CPU 30准备从接收FIFO存储器50读取存储的DLPDU数据时,经由寄存器60的寄存器RegID,CPU中的软件生成GetDataByte_in命令。任何从寄存器60的寄存器RegID中的读取将来自接收FIFO寄存器的数据立即放置在数据总线上以供CPU 30读取。将GetDataByte_in信号发给所有三个接收消息对象,但是一次三个接收消息对象中只有一个对于CPU 30是可读取（或可视）的。EventMUX_sel (2:0)信号经由MUX 106,选择当前可视的接收消息对象以供CPU 30读取。例如,如果RcvMsg1_visible是有效的,则通过接收消息对象rcvmsg1生成GetDataByte1。GetDataByte1、GetDataByte2和GetDataByte3作为输入被提供给OR门114,其输出是全局GetDataByte信号。当激活全局GetDataByte信号时,接收FIFO存储器50的读取指针122递增一个位置。当从rcvmsg1执行读取操作时,将读取指针122与rcvmsg1的结束位置值相比较。这使得CPU 30知道对于与rcvmsg1相关的消息,接收FIFO存储器50中的字节数量。

[0100] 前端状态机46将串行数据流解码,并将其转换为8比特并行格式的字节。在形成所述字节后,前端状态机46创建写入脉冲（经由RcvDataByte),其将已编码数据存储至由写入指针所指向的接收FIFO存储器50中的位置。写入脉冲使写入指针120递增,以准备写入下一个DLPDU字节。在将有效接收消息的DLPDU数据写入接收FIFO存储器50中时,接收FIFO存储器50中的写入指针的当前值被连续地传送到对应接收消息对象的结束位置值。由于以和将接收消息对象加入接收消息对象队列100相同的顺序将DLPDU数据加入接收FIFO存储器50,所以在接收FIFO存储器50和接收消息对象队列100之间保持一致的排序。

[0101] 接收FIFO存储器50中存储的数据包含多达3个完整的接收消息的DLPDU数据（总体多达63字节）。每个接收消息对象包含用于存储可由软件读取的与有效接收消息的接收状态有关的若干属性（如由MsgQmngr 80选择的）。这些属性包括上溢、下溢、数据准备好、消息结束位置(EndPosition)和有效标记。

[0102] 上溢是Boolean属性,当设定它时,指示在存储来自分段12的接收消息的DLPDU数据时发生接收FIFO存储器50的上溢。如果在由CPU30从接收FIFO存储器50读取DLPDU数据之前接收FIFO存储器50满了,则接收FIFO存储器50向所已接收消息对象激活指示已经发生上溢条件的信号。这将防止在从接收FIFO存储器50中读取一个DLPDU数据字节之前,再将数据写入接收FIFO存储器50。当上溢条件发生时,针对可视的接收消息对象,设定上溢属性。当具有上溢属性设置的接收消息对象对于CPU 30可视以供其读取时,由CPU 30读取并处理上溢条 件。CPU 30通过向寄存器Reg16的合适比特进行写入而确认设置上溢属性（并且随后清除上溢属性）。

[0103] 下溢是Boolean属性,其指示由于缺乏数据最后一次对接收FIFO存储器50进行读取的尝试失败。这发生在当接收FIFO存储器50为空（即没有DLPDU字节供读取）时CPU 30试图产生GetDataByte_in信号的时候。如果发生了上述情况,则将不增加读取指针

122，因此保持读取指针 122 的合适定位。当发生下溢条件时，针对可视的接收消息对象，设定下溢属性。CPU 30 通过向寄存器 Reg16 的合适比特进行写入，确认设置下溢属性（并且随后清除下溢属性）。

[0104] 对于所有接收消息，当从前端状态机 46 发送数据以写入接收 FIFO 存储器 50 时，接收 FIFO 存储器 50 连续地监视它是否接近它的 63 比特存储限制。数据准备好是可配置的 Boolean 属性，它指示接收 FIFO 存储器 50 接近满的状态，并且需要由 CPU 30 对其进行读取以防止接收 FIFO 存储器 50 的上溢。如果接收 FIFO 存储器 50 中的未读取 DLPDU 数据字节的数量等于或大于数据准备好阈值，则在有效的接收消息对象中生成 IRQ。当接收消息对象对于 CPU 30 可视以供其读取时，CPU 30 可以读取并处理所述数据准备好属性。CPU 30 通过对寄存器 Reg16 的合适比特进行写入，确认设置数据准备好属性（并随后清除数据准备好属性）。

[0105] 结束位置是整数属性，它在为当前有效消息写入 DLPDU 数据时存储接收 FIFO 存储器 50 的写入指针 120 的位置。当写入指针 120 递增时，持续地更新当前有效接收消息对象的接收位置属性。当从接收 FIFO 存储器 50 读取 DLPDU 数据字节时，读取指针 122 递增。当读取指针 122 的值等于结束位置值时，对于当前可视的接收消息对象，在接收 FIFO 存储器 50 中未存储其它数据。当对于可视接收消息对象没有可获得的数据时，设定空标记，以防止生成可视接收消息对象的 GetDataByte 输出。这有助于保持读取指针 122 的合适定位和 DLPDU 数据关联整体性。每个接收消息对象的结束位置属性向 MUX 112 提供输入，并且可视接收消息对象的结束位置属性值被发给读取指针 122。

[0106] 有效标记是 Boolean 属性，其指示还（经由信号 RcvMsg_sel）选择由 CPU 30 正在处理的接收消息对象（即可视接收消息对象）。当针对接收消息检测到 EOA 时，有效标记返回低。当从分段 12 接收到消息时， 当前所选的接收消息对象更新它的相关消息数据。例如，如果当 MAU 38 接收消息时在接收 FIFO 存储器 50 中不包含 DLPDU 数据，则由 MsgQmngr 80 设定信号 RcvMsg1_sel，因此在接收消息对象中 rcvmsg1 中激活有效标记属性。当针对接收消息对象设定了有效标记时，将写入指针 120 的当前位置存储在所选接收消息对象的结束位置属性中。

[0107] 对于某些接收消息，在将消息的所有 DLPDU 数据存储在接收 FIFO 存储器 50 中之前，CPU 30 可以确定是否有必要处理整个消息。例如，假设分段 12 上的 20 字节的消息正在被接收的进程中，并且接收消息对象 rcvmsg1 被选择 (RcvMsg1_sel ＝ 1) 并且是可视的 (RcvMsg1_visible ＝ 1)。在将一些 DLPDU 字节写入接收 FIFO 存储器 50 中之后，CPU 30 可以读取寄存器 Reg1F，并且针对已经写入接收 FIFO 存储器 50 中的 rcvmsg1，确定 DLPDU 字节的数量。从寄存器 Reg1F 读取的值是 rcvmsg1 的结束位置属性值与读取指针 122 的值的比较。然后，CPU 30 可以经由寄存器 Reg1D，读取存储在接收 FIFO 存储器 50 中的部分消息。由所述部分消息，CPU 30 可以确定不需要对余下的字节进行处理，并且可以丢弃（即过滤）消息的余下部分。在这一点上，CPU 30 通过对寄存器 Reg15 执行写入操作，发起 Event Ack 信号。所述 Event_Ack 信号通过 MUX 108 生成 CurMsg_Ack 信号。

[0108] Event Ack 信号的发起产生要发生以为下一接收消息准备接收 FIFO 存储器 50 的事件的次序。首先，当生成 CurMsg_Ack 信号时，将通过 MsgQmngr 80 禁用信号 RcvMsgQMux_enb。当 RcvMsgQMux_enb 信号为低时，将忽略任何在有效接收消息对象的输入处发生的附

加事件。CPU30 保存读取接收消息的余下部分的处理时间，这使 CPU 30 能够在设备 24 中执行其它处理。然后，由相应的 RcvMsg_Ackn 信号（如接收消息对象 rcvmsg1 的 RcvMsg_Ackn）清除针对有效接收消息对象所设定的属性状态（如上溢、下溢等）。随后，CurMsg_Ack 信号禁用指针使能逻辑 118，指针使能逻辑 118 控制接收 FIFO 存储器 50 的写入指针 120 的递增。该行为防止将与有效接收消息对象相关的有效接收消息的附加 DLPDU 字节写入接收 FIFO 存储器 50。由前端状态机 46 对余下的 DLPDU 数据进行解码，但并不存储。CurMsg_Ack 信号还使 CurPosition116 的当前值复制到写入指针 120。RcvMsg_Ack 信号（从 OR 门 110 输 出）启动 MUX 112 输出的有效接收消息对象的结束位置值的采样（例如接收消息对象 rcvmsg1 的结束位置 1），并且随后启动向读取指针 122 复制所采样的结束位置值。因此，写入指针 120 和读取指针 122 被重新定位至相同值。

[0109] 当部分接收的已丢弃的消息发生活动结束 (EOA) 事件脉冲时，接收消息对象队列 100 中的下一接收消息对象的 RcvMsg_sel 信号变得有效。当在分段 12 上接收新消息时，将它写入接收 FIFO 存储器 50。将下一接收消息对象的 DLPDU 数据的第一字节存储在写入指针 120 的新的重定位值处。结果，由新消息盖写了已丢弃的消息的部分接收部分。

[0110] 总之，消息过滤向硬件提出了特殊问题，因为消息过滤器在决定是否对消息进行过滤之前需要部分地接收消息。如果软件消息过滤决定对消息的余下部分进行过滤，则需要丢弃部分接收的消息。本发明是一种系统和方法，用于从接收数据存储器中丢弃部分接收的消息。当接收消息时，激活接收消息对象，而且跟踪接收数据存储器中的写入指针 120 的当前位置，并将其存储在有效接收消息对象中。然后从写入指针 120 的初始位置处开始，将所述接收消息写入接收 FIFO 存储器 50。当向接收 FIFO 存储器 50 写入消息时，写入指针 120 的位置递增。如果 CPU 30 确定将忽略所述消息，则禁用有效接收消息对象，因此忽略与所述消息有关的余下事件。此外，禁用接收数据存储器，因此防止进一步向接收 FIFO 存储器 50 写入与被忽略消息相关的 DLPDU 数据。当忽略所述消息时，将读取指针 122 定位在接收 FIFO 存储器 50 中的当 CPU 30 决定过滤余下消息时的点处。然后优选地，将写入指针 120 重新定位至接收 FIFO 存储器 50 中与读取指针 122 相同的位置。

[0111] 在针对消息接收到所有 DLPDU 数据之前，通过使软件软件忽略消息，为 CPU 30 节约了宝贵的处理时间。此外，由于接收 FIFO 存储器 50 可能包含多达三个接收消息对象的 DLPDU 数据，所以重要的是，使硬件适当地跟踪和控制写入指针 120 和读取指针 122 的定位，以保持 DLPDU 数据与适当的接收消息对象的关联。在 CPU 处理时间十分关键的情况下，使写入指针 120 和读取指针 122 正确对准的重要性显得尤为明显，例如当丢弃具有大量与它相关的 DLPDU 数据的接收消息时，或者在对于低功率应用以较低时钟进行操作的 CPU 中。

[0112] 虽然已经参照优选实施例描述了本发明，但是本领域技术人员将认识到可以在形式和细节上进行改变，而不脱离本发明的精神和范围。

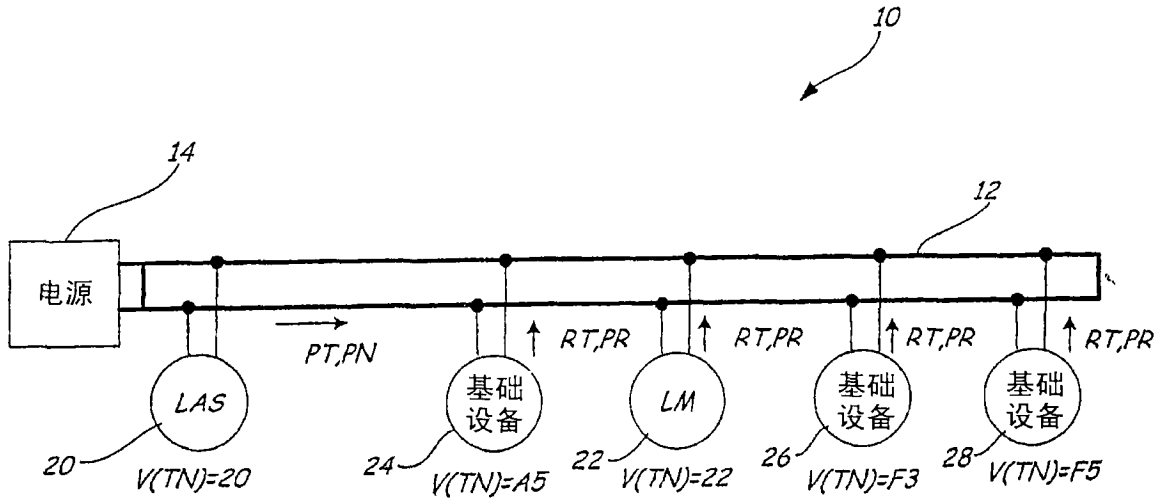

10
14
12
电源
PT,PN
RT,PR
RT,PR
RT,PR
RT,PR
LAS
20
V(TN)=20
基础设备
24
V(TN)=A5
LM
22
V(TN)=22
基础设备
26
V(TN)=F3
基础设备
28
V(TN)=F5

图 1

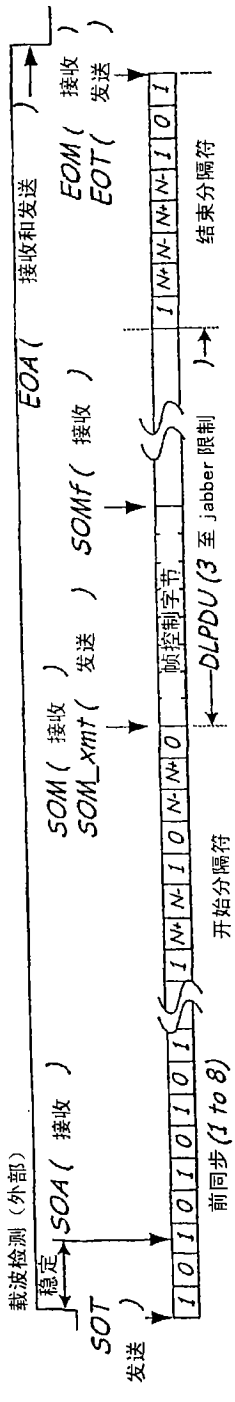

载波检测（外部）
稳定
SOT
发送
SOA（ 接收 ）
前同步(1 to 8)
SOM（ 接收 ）
SOM_xmt（ 发送 ）
开始分隔符
帧控制字节
DLPDU(3 至 jabber 限制
SOMf（ 接收 ）
EOA（ 接收和发送 ）
EOM（ 接收 ）
EOT（ 发送 ）
结束分隔符

图 2

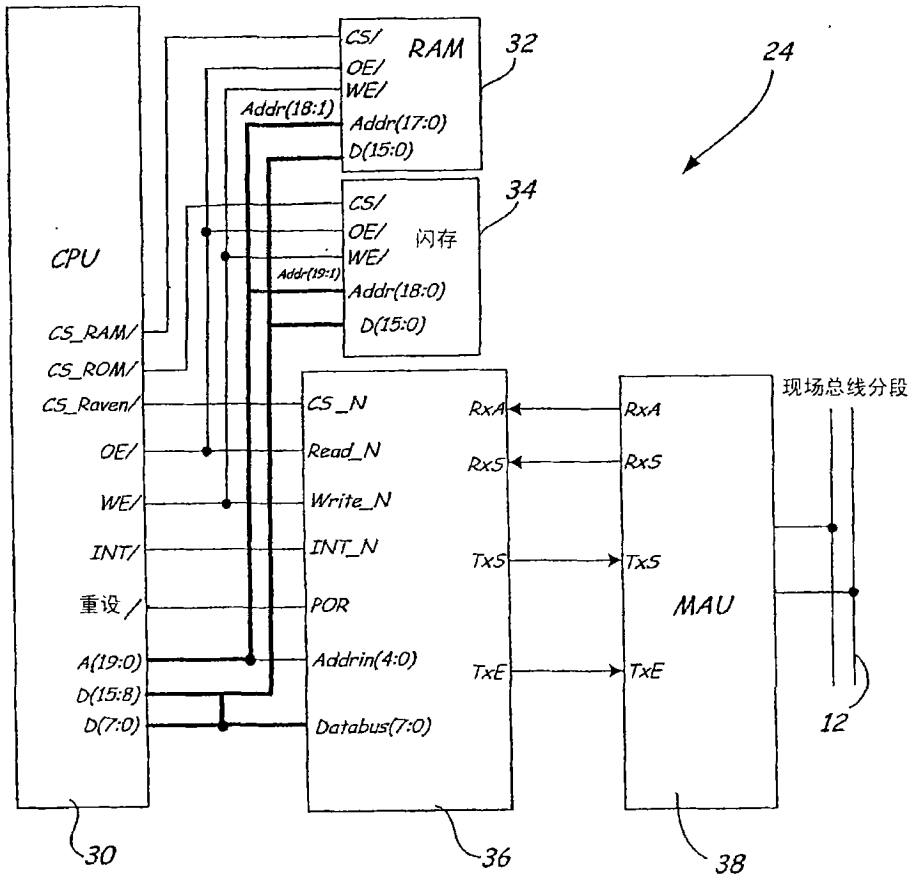

CPU
CS_RAM/
CS_ROM/
CS_Raven/
OE/
WE/
INT/
重设/
A(19:0)
D(15:8)
D(7:0)
30
CS/
OE/
WE/
Addr(18:1)
Addr(17:0)
D(15:0)
RAM
32
CS/
OE/
WE/
Addr(19:1)
Addr(18:0)
D(15:0)
闪存
34
24
CS_N
Read_N
Write_N
INT_N
POR
Addrin(4:0)
Databus(7:0)
36
RxA
RxS
TxS
TxE
RxA
RxS
TxS
TxE
MAU
38
现场总线分段
12

图 3

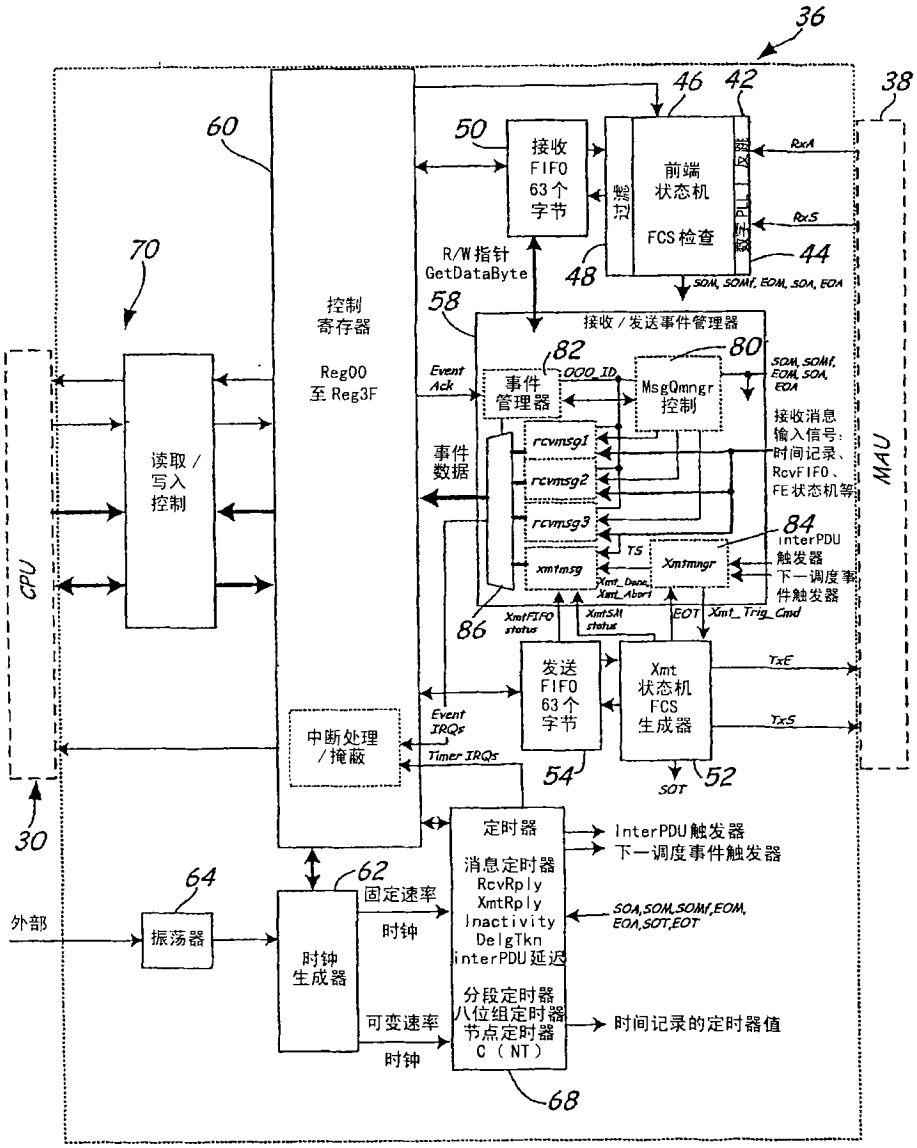

36
38
46
42
50
60
70
58
82
80
84
86
54
52
64
62
44
48
接收
FIFO
63个
字节
前端
状态机
FCS检查
RxA
RxS
R/W指针
GetDataByte
SOM, SOMf, EOM, SOA, EOA
控制
寄存器
Reg00
至Reg3F
接收/发送事件管理器
Event
Ack
事件
管理器
000_ID
MsgQmngr
控制
SOM, SOMf,
EOM, SOA,
EOA
接收消息
输入信号:
时间记录、
RcvFIFO、
FE状态机等
事件
数据
rcvmsg1
rcvmsg2
rcvmsg3
xmtmsg
TS
Xmtmngr
InterPDU
触发器
下一调度事
件触发器
Xmt_Done,
Xmt_Abort
XmtFIFO
status
XmtSM
status
EOT
Xmt_Trig_Cmd
读取/
写入
控制
CPU
发送
FIFO
63个
字节
Xmt
状态机
FCS
生成器
TxE
TxS
SOT
中断处理
/掩蔽
Event
IRQs
Timer IRQs
30
定时器
InterPDU触发器
下一调度事件触发器
消息定时器
RcvRply
XmtRply
Inactivity
DelgTkn
InterPDU延迟
SOA,SOM,SOMf,EOM,
EOA,SOT,EOT
分段定时器
八位组定时器
节点定时器
C (NT)
时间记录的定时器值
外部
振荡器
时钟
生成器
固定速率
时钟
可变速率
时钟
68
MAU

图 4

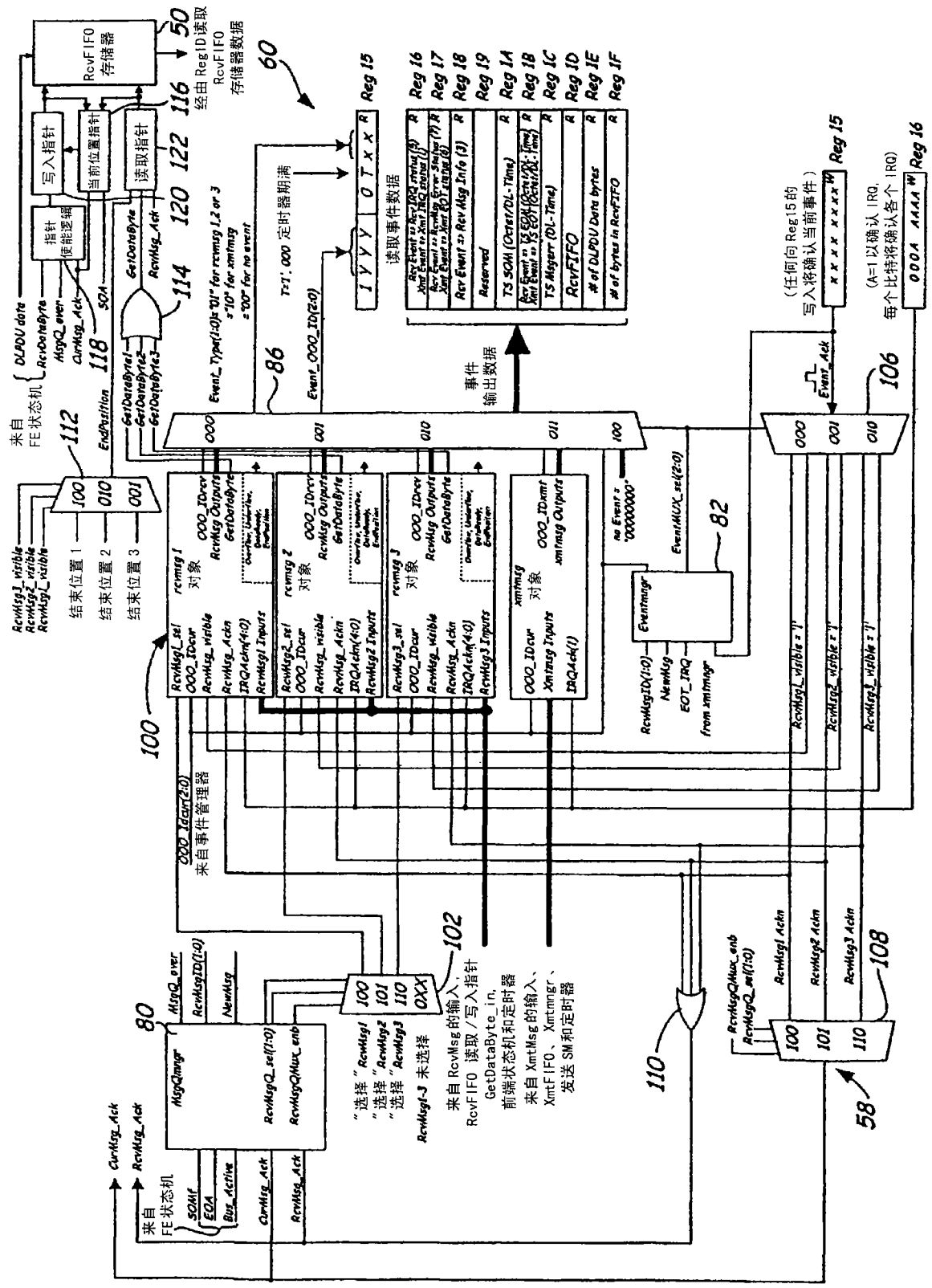

RcvFIFO 存储器
写入指针
当前位置指针
读取指针
指针 使能逻辑
经由 RegID 读取 RcvFIFO 存储器数据
来自 FE 状态机
结束位置 1
结束位置 2
结束位置 3
读取事件数据
事件 输出数据
来自事件管理器
来自 RcvMsg 的输入、RcvFIFO 读取 / 写入指针 GetDataByte_in、前端状态机和定时器
来自 XmtMsg 的输入、XmtFIFO、Xmtmngr、发送 SM 和定时器
(任何向 Reg15 的写入将确认当前事件)
(A=1 以确认 IRQ, 每个比特将确认各个 IRQ)

图 5