收稿日期：

基金项目：上海市自然科学基金项目(15ZR1429300)

作者简介：赵国承（1998-），男，上海理工大学光电信息与计算机工程学院硕士研究生，研究方向为汽车电子、现场总线、嵌入式系统；; 张凤登（1963-），男，博士，上海理工大学光电信息与计算机工程学院教授、博士生导师，研究方向为分布式系统、过程控制、现场总线。

避免饥饿的CAN总线优先级反转方案

赵国承1，张凤登2

（1. 上海理工大学 光电信息与计算机工程学院，上海 200093；2．上海理工大学 光电信息与计算机工程学院，上海 200093）

摘要：在CAN总线中，当两个或更多节点同时开始发送时，存在基于优先级的仲裁方案来决定哪个节点将被授予继续发送的许可。在总线仲裁期间，每一个发送节点时刻监控总线状态，同时将接收到的位与发送的位进行比较，如果在发送隐性位1时接收到了显性位0，则该节点会停止发送。因此，基于优先级的策略可能会导致饥饿问题，即高优先级的发送请求总是阻止低优先级的发送请求。在本研究中，提出了一种避免饥饿的优先级反转方案，能够在一定程度上改善优先级高的消息持续占用总线的问题。运行仿真结果表明,使用优先级反转方案之后,不仅减少了低优先级报文的平均响应时间，而且还提高了低优先级报文的到达率,未出现优先级死锁现象，证明了算法有效性。

关键词：优先级反转；CAN总线；优先级队列；响应时间

0引言

CAN总线是用于连接电子控制单元（Electronic Control Unit，ECU）的多主机广播串行总线标准，每个节点都能够发送和接收消息，但不能同时发送和接收消息[1]。在CAN总线中，来自多个节点的高优先级消息持续发送时会阻塞低优先级消息占用CAN总线，造成低优先级消息饥饿[2]。

近年来，已经有研究使用一些方案来解决CAN总线饥饿问题。文献[3]中，Murtaza和Khan提出在CAN总线上增加一个主节点，以防止其他低优先级节点出现阻塞问题。主节点必须尝试检测饥饿节点，并确保饥饿节点保持参与通信。主节点始终监视CAN网络，并在收到来自所有节点的消息时保持沉默。如果主节点发现来自某些节点的消息没有参与通信，则该主节点将变为活动状态，这确保了来自这些节点的消息可以在下一轮仲裁中进行访问以进行广播。虽然该方案可以防止启动问题，但由于需要额外的节点支持，因此该方案需要更高的负担。文献[4]中,提出基于使用优先级和FIFO队列的CAN总线消息的调度研究，但其只关注了可行调度表的生成却因此忽略了CAN总线中存在的饥饿问题。文献[5]中开发了一种无饥饿的方案，当存在任务发送失败的次数大于设置的阈值的时候，会使用优先级反转将该任务反转为最高的优先级。文献[6]中提出了一种新的无饥饿的CAN总线仲裁模型，其特点是利用消息标识符的最高两位动态实现优先级上升。文献[7]中，Kimm等人提出了一种基于RM算法的避免饥饿CAN总线调度方案，能够将仲裁中失败的消息优先级反转为更高的优先级。

但是以上这些工作大多过分重视将优先级低的任务在CAN总线仲裁中动态变为高优先级任务，而本文则是更多的考虑将优先级高的任务在一定条件下进行优先级反转，动态降低其优先级，从而能够有效的避免CAN总线造成的饥饿问题。

第1节介绍了CAN总线帧格式和仲裁方法，第2节提出并实现了避免饥饿的优先级反转算法，第3节进行实验仿真验证，第4节为结语。

1.CAN总线帧格式和仲裁方式

自从德国Bosch公司在汽车工程师协会(SAE)大会上引入串行总线系统以来，CAN总线已经逐渐成为最成功的数据通信协议之一[8]。CAN总线中采用载波监听多路访问／冲突避免（CSMA/CA）协议[9,10]。所以CAN总线的访问是无冲突的、按位仲裁的。无冲突意味着总是更高优先级的消息在仲裁中获胜并发送，在发送过程中其它消息禁止发送。按位仲裁可以理解为，如果有两个或多个节点同时进行传输，且有显性位0和隐形位1同时发送，规定显性位0的权重大于隐形位1，故显性位0可以覆盖隐形位1。

在CAN总线当中有四种不同的帧，包括数据帧、远程帧、错误帧和过载帧。其中数据帧能够将数据从发生端广播至接收端，数据帧格式如图1所示，携带有标识符的远程帧用来请求另一节点发送具有相同标识符的数据帧。错误帧发送用来指示总线中发生错误。当有数据帧与相邻的一个数据帧或远程帧之间出现额外延迟时，过载帧将会被发送。CAN总线上任何一个节点都能够在CAN总线上发送消息，无需获得总线上其他节点的许可。

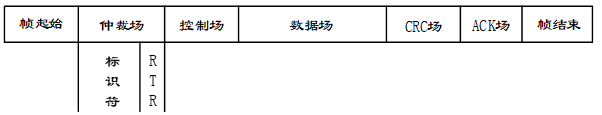


图1 数据帧格式

数据帧分为两种格式，标准格式以及扩展格式。其中标准格式的标识符长度为11位，而扩展格式标识符具有29位。本文以标准格式为进行论述，对数据帧中仲裁场里面的标识符进行进一步的分析研究。

当两个或多个节点同时开始传输，通过对总线进行监控，可以确定每个节点当前正在传输的报文优先级高低，若在仲裁中某一节点胜出，那么该节点继续传输其报文的剩余部分，此时其他所有仲裁失败节点均退出并继续等待下一次仲裁开始。同时因为CAN总线没有全局时钟，所以需要节点在每次消息传输时重新同步，而且每个节点必须与首先开始传输的节点引起的帧起始上升沿同步，以便总线仲裁能够正常工作。

2 优先级反转算法设计

2.1 算法设计

在本算法中，11位消息标识符被设计为三部分，分别是1位动态反转位、1位级别位和9位静态位，如图2消息标识符ID划分所示。当然这仅仅是在标准格式中的划分，除此之外也可以应用在扩展格式中。静态标识位（ID0-ID8）从最低的111111111开始，以最高的0位000000000结束。级别标识位ID9用来区分高优先级组和低优先级组，两组分别含有10个报文，其中高优先级组的级别标识位为0，低优先级组的级别标识位为1。动态反转位ID10默认为0未反转，当高优先级组满足反转条件时，会动态反转为1。

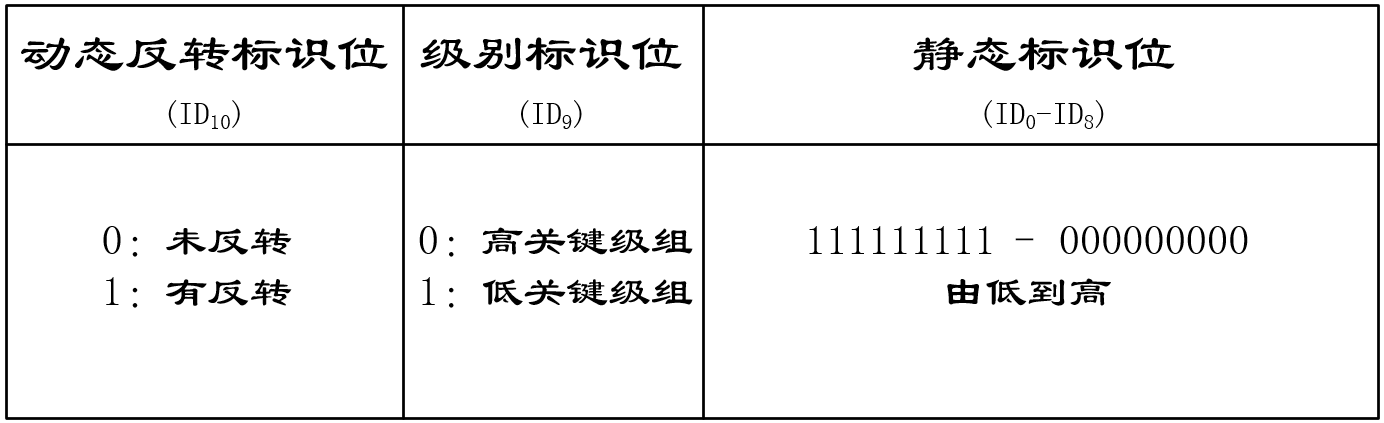


图2 消息标识符ID划分

CAN总线上的发送的报文都是由其节点生成的，故当我们为不同的报文分配优先级时，这些报文的优先级将伴随着报文同样保存在节点中。当其中的报文被节点发送时，节点将以其最初设置的优先级进行发送。由于CAN总线是工作在基于固定优先级的消息调度上，每当CAN总线上的节点生成报文时，该报文所获得的优先级将一直被固定不变，而且每一个报文的优先级都是独一无二的，不能出现重复的优先级，否则很有可能造成死锁。

对于报文的优先级分配，本文选择了非抢占式的单调速率（RM）算法，之所以选择非抢占式的RM算法是因为两点：（1）如图3所示，非抢占式的调度方法可以确保一旦发送某一消息，能够一直发送直到本次发送完毕，在此期间不能被其它高优先级的消息打断。（2）RM算法是根据报文的周期来分配一个固定优先级，其特点是，报文的发送周期越短则其优先级越高，反之报文的发送周期越长则其优先级越低，要求在报文开始发送之前便分配完毕且运行过程中始终固定。



图3 非抢占式调度

在本文之前也有一些研究建立了避免饥饿的模型，但是大多数都是动态的将低优先级转变为高优先级，但是这种方法没有考虑到，如果CAN总线上存在非常重要的报文需要立刻发送时，很有可能需要经过几轮仲裁才能发送。而本文中的模型则不会出现这一问题。

算法的实现如下所述，在CAN总线中，当高优先级组在一定的时间内出现某一报文发送的次数大于设置的固定值，则在下一次其继续参与CAN总线仲裁之前，将其动态反转位ID10进行反转由0变为1,从而使其比低优先级组里面的报文优先级还低，此时处于饥饿中的报文得到机会发送，实现避免饥饿的目的。此时，当其动态反转位变为1时，并不意味着其丧失本次继续参与CAN总线仲裁的资格，而是会进一步的判断当前CAN总线是否处于“满员”的状态，如果总线空闲，则该反转后的报文正常发送，否则不会发送。这样可以避免当总线阻塞时，反转后发送的消息经过多次仲裁都不能占用总线的问题，同时还可以防止报文发送的顺序错乱。最后不管本次是否发送成功，都会在下一次参与CAN总线仲裁之前将其动态反转标识位变为0。有了1位动态反转标识位，任何想要持续占用总线的高优先级组报文，在持续占用一定次数后都会被中断。研究中使用的每个11位ID的动态反转位开始均为0，级别标识位可在0与1之间进行划分，静态标识位内数字任意。

2.2 任务模型

模型如图4所示，在本模型中，设被调度的报文组{M1,M2，…，Mn}，Mi是其中的一个报文，i=1，2，… n。避免饥饿的优先级反转模型主要由2个参数决定Ttc和TMax，其中Ttc代表检测报文发送次数的时间，TMax代表在Ttc时间内发送报文的次数。同时进行了如下定义：

定义1：报文Mi的传输周期等于其截止时间

其中为报文Mi传输周期，为报文Mi截止时间。

定义2：CAN总线利用率U定义为

其中K为报文Mi执行时间，P为报文Mi传输周期，i=1，2，… n。

定义3：报文Mi的响应时间T定义为

其中t1为报文Mi的激活时间，t2为报文Mi的发送完成时间。

则平均响应时间定义为

其中n为报文数量，i=1，2，… n。

定义4：到达率S定义为

其中R指的是一段时间内实际发送报文的数量，C为预计发送报文的数量。



图3 避免饥饿的优先级反转模型

其中算法所用的部分代码如图4所示，在这部分代码中实现了检测报文发送次数防止遗漏、到达规定次数以后进行反转以及未反转报文的发送功能。

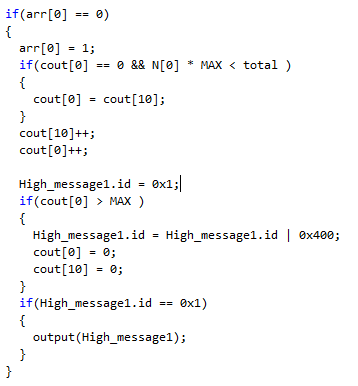


图4 优先级反转算法部分代码

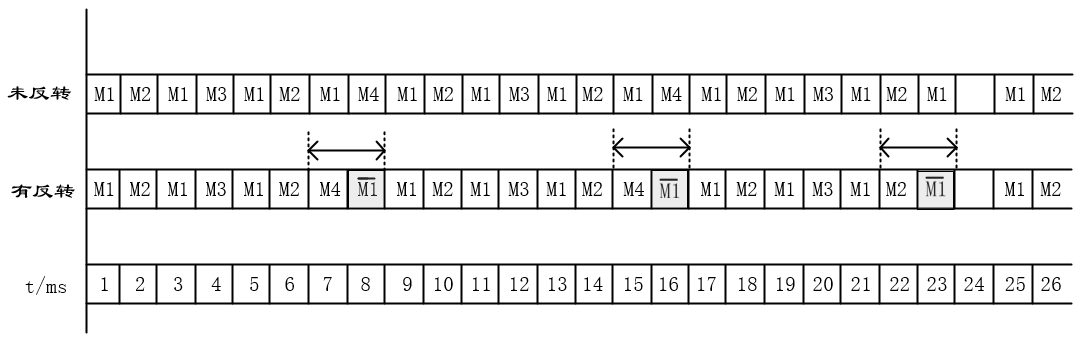
3 实验仿真

在本节中，我们使用CANoe平台来评估优先级反转方案的性能，利用CANoe平台中的CAPL语言来进行编程。规定实验开始所有报文均在0时刻释放，设置了两组周期消息队列，分别为高优先级组和低优先级组，两组报文的截止日期均等于报文的发送周期，实验先将每组设置为2个报文。

其中高优先级组的两个报文为M1和M2，低优先级组的两个报文为M3和M4，为了进一步的验证避免饥饿的优先级反转模型与正常固定优先级相比的优势，实验将分为三组分别为CAN总线正常工作、CAN总线满利用率工作以及CAN总线产生饥饿时的工作进行分析。

3.1 CAN总线正常工作时

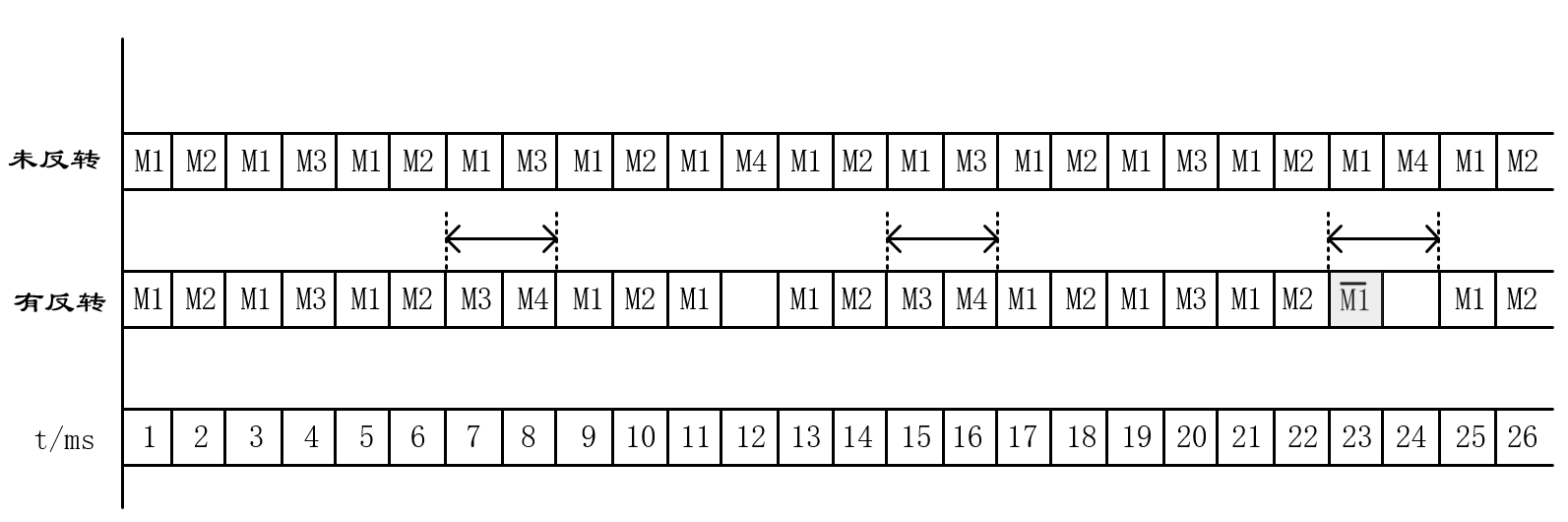
设报文M1的发送周期为2ms，优先级设置为0x01，报文M2的发送周期为4ms，优先级设置为0x02，报文M3的发送周期为8ms，优先级设置为0x201，报文M4的发送周期为12ms，优先级设置为0x202。规定两组报文的执行时间K均设置为1ms，检测时间Ttc设置为8ms，检测报文发送次数TMax设置3。如图所示，通过对比未进行反转的固定优先级调度和有反转的非抢占式RM调度在CAN总线的发送队列，可以知道，未进行反转的固定优先级调度和有反转的非抢占式RM调度虽然均能够将报文正常发送，但是使用了反转的非抢占式RM调度能够较好的降低了低优先级组报文的响应时间T，例如报文M4。其中反转报文是报文M1的反转，反转后，根据避免饥饿的优先级反转模型可知，其优先级反转为0x401，远远小于低优先级组的两个报文的优先级。因此反转后报文M4可以在反转报文的前面执行，当然因为CAN总线不存在饥饿的情况，故低优先级组的到达率S始终为1。



图

3.2 CAN总线满利用率工作时

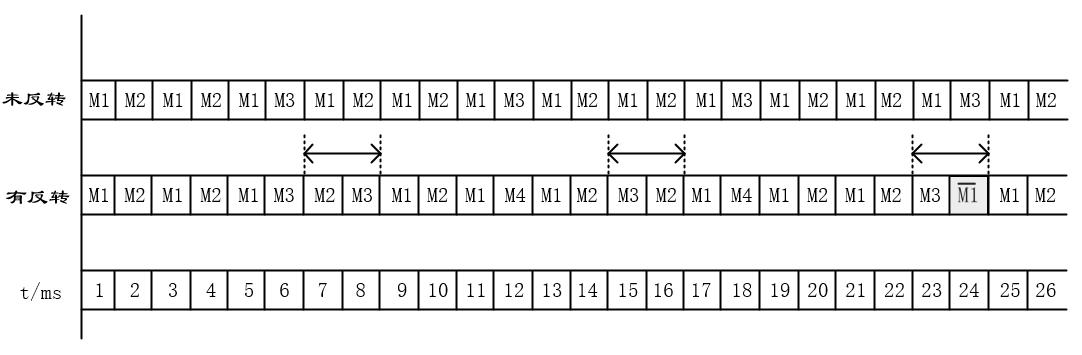
在此实验中除了将报文M3的发送周期为6ms以外，其它所有的实验参数均与CAN总线正常工作时一样。通过观察图可知，此时未进行反转的固定优先级调度恰好能够将所有报文在其周期内发送，而此时有反转的非抢占式RM调度则很明显的降低了低优先级组报文的响应时间T，此时CAN总线没有饥饿的情况，故低优先级组的到达率S始终为1。



图

3.3 CAN总线产生饥饿时的工作

本次将报文M2的发送周期改为3ms，报文M3的发送周期为改6ms，其它所有的实验参数均与CAN总线正常工作时一样。如图可知，未进行反转的固定优先级调度在实验中产生了饥饿，导致低优先级组的报文M4一直没有占用总线的机会，使其一直处于饥饿当中，故低优先级组的到达率S小于1。而此时有反转的非抢占式RM调度则能够避免CAN总线饥饿问题的产生，使得两组报文在其周期内均能够得到占用总线的机会。此时低优先级组报文的响应时间T得到了明显的降低，由于CAN总线没有产生饥饿问题，故低优先级组的到达率S仍然为1。



图

3.4 扩大仿真

为了更进一步的验证避免饥饿的优先级反转模型与正常固定优先级相比的优势，将实验中的报文数量扩大，设置报文数量n为20，每组报文数量设置为10，仿真时间为1s，对比低优先级组在未进行反转的固定优先级调度和有反转的非抢占式RM调度下的到达率和平均响应时间。

对于其平均响应时间，如图5所示，分别验证了在不同的CAN总线利用率下有反转的非抢占式RM调度和无反转的CAN总线正常调度对低优先级组平均响应时间的影响。当CAN总线利用率逐渐增加时，不管低优先级组是处在未进行反转的固定优先级调度还是有反转的非抢占式RM调度，其平均响应时间都在不断的增加，但是有反转的非抢占式RM调度的平均响应时间总是小于未反转的CAN总线正常调度的平均响应时间。

图5 低优先级组有无反转的平均响应时间

对于其到达率，在本实验中，如图6所示，分析了CAN总线由正常到产生饥饿时，有反转的非抢占式RM调度和无反转的CAN总线正常调度对低优先级组到达率的影响。可知若CAN总线的利用率小于等于100时，有无反转对低优先级组的到达率并无显著影响，当CAN总线的利用率大于100时，此时CAN总线处于饥饿状态。而且当CAN总线的利用率逐渐增加时，无反转的CAN总线正常调度下的低优先级组到达率持续减少，直到为0，此时低优先级组中没有一个报文能够在占用一次CAN总线，与之相反的是，在有反转的非抢占式RM调度下的低优先级组到达率则基本维持不变。

图6 低优先级组有无反转的到达率

4 结论

本文针对CAN总线中存在的饥饿问题，建立了一个优先级反转模型，如图3所示，与以往不同的是，文中的模型在满足优先级反转条件下，会使高优先级反转为低优先级。通过进行试验仿真，在不同的CAN总线利用率下，该模型能够通过改善CAN总线中周期任务的平均响应时间来减少饥饿。当CAN总线利用率超过100时，有反转的低关键级组到达率也能够保持正常，这表明我们的避免饥饿优先级反转方案工作良好。

参考文献

1. 张凤登.现场总线技术与应用[M].北京：科学出版社，2008.
2. Anwar K, Khan Z A. Dynamic priority based message scheduling on controller area network[C]//2007 International Conference on Electrical Engineering. IEEE, 2007: 1-6.
3. Murtaza A F, Khan Z A. Starvation free controller area network using master node[C]//2008 Second International Conference on Electrical Engineering. IEEE, 2008: 1-6.
4. Davis R I, Kollmann S, Pollex V, et al. Schedulability analysis for Controller Area Network (CAN) with FIFO queues priority queues and gateways[J]. Real-Time Systems, 2013, 49(1): 73-116.
5. Lin C M. Analysis and modeling of a priority inversion scheme for starvation free controller area networks[J]. IEICE transactions on information and systems, 2010, 93(6): 1504-1511.
6. Kimm H, Kimm H. Modeling and Verification of Starvation-Free Bitwise Arbitration Technique for Controller Area Network Using SPIN Promela[C]//International Conference on Ubiquitous Information Management and Communication. Springer, Cham, 2019: 195-210.
7. Kimm H, Chelladurai J. Starvation-avoidance CAN Scheduling for Shorter Worst-case Response Time with Priority Queues[C]//2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM). IEEE, 2020: 373-378.
8. 张凤登.分布式实时系统[M].北京：科学出版社，2014.
9. Lawrenz W. Can System Engineering: From Theory to Practical Applications[M]. DBLP, 2007.
10. Bosch R G. CAN specification version 2.0: Robert Bosch GmbH[J]. Systems und Products for Car Manufacturer, 1991.