基于CAN/Flexray的红外线测温设计

想法：在原基础上加一个标志位以及利用标识符ID的最高位来进行位反转。

多重标志位优先级反转来确保消息发送

或者加上一个等差数列或其他来进行等级的提升

如果高优先级的任务超过一定的次数发送，则会让其停止一次仲裁，采用间隔（每两次参与一次或者其他）的方法来仲裁，能够保证低优先级得到利用。

分为两部分一部分是重要（紧急）的部分，另一部分次重要部分，其中次重要部分采用了动态FQ来做，但是如果重要部分一直持续参与仲裁的次数超过n，则执行反转。反转的触发条件可以是或者等比的方式减少ID，增加优先级

或者反转高优先级，使其优先级最低

能够确保具有紧急非周期重要信息能够得到及时的发送

如果重要组能够接替进行，不是一个重要节点一直发送，而是几个重要节点一直交替发送，那么这种方法的效果则不明显

称为优先级退化

第一位用来调节优先级

第二位用来分组

高组00\*\*\*\*\*\*\*\*\*

低组01\*\*\*\*\*\*\*\*\*

两组，每组十个信号，

四组，每组四个信号，

由于进行了分组且每次优先级的反转都是在最高位，故不会产生死锁现象。

按某一键之后，周期发生变化

当设置为8位数据传送的时候，通常每一毫秒都可以发送四次的报文，故根据两毫秒或者一毫秒为判断时间，只要一毫秒内发送过的都给予计数，连续发送几毫秒的则给一个优先级转换，将其变为小的优先级，并停止发送一次，

设置正常组和问题组以及方案组，分别实验得出这种方法的优势

**在一定时间内，满足发送一定次数的报文则被优先级反转0-1**

**停一下暂时不发，等一会发，理解为如果当前CAN总线时拥挤的则不发，如果满足空闲的条件则发送。**

画一个流程图

问题：高优先级反转以后要不要发送，如果发送的话一直发送不过去，或者说能发送过去但是时间上已经很迟，远远落后下一次未反转的消息怎么办

方法：在重新反转为高之后第一次发送之前如果81还没有发送出去，则重新发送

问题 上一次没有发送完的会继续延迟到下一次并且不能被下一次计数器简单计算到

利用一个数把每次超过的数量储存下来

如果只是单纯的高优先级反转，则会出现反转到低优先级以后，发送的消息被阻塞，这时如果下一次再反转回高优先级以后，有很大的概率高优先级发送几次报文以后，反转到低优先级的报文还没有发送出去，这样就会产生很大的问题。本来是为了中断高优先级的消息持续阻塞总线的问题，如果只是单纯反转，有极大概率出现发送顺序错乱以及传递时间过长的问题，那样的话就得不偿失。故本文提出一种方法，在报文发送时设置一个标志位时刻来检测总线上排队发送的报文数量，如果在高优先级反转到低优先级时，此时总线在本次不阻塞则发送，否则停止本次发送，等待优先级变为高后在发送。

分析一下参与的次数

测试到达率以及负载率

分析100ms，500ms， 1s， 5s, 10s低优先级消息的到达率

也可以接着分析

满负载的时候可以改变到达率

非的话可以改变负载率

3.23 问题

解决如果极端情况低优先级发送三个的时候，反转优先级两个也同时发送情况，但是这种概率在本实验很低可以忽略。

3.24

问题

可以设置一个标志位来检测到了第三次和第四次的空格期间继续发送，但是有一个问题就是虽然克服了只发送一次的问题，但是很有可能会造成虚假的阻塞。

3.25问题

产生了阻塞，因为计算发送数量出现问题

到达率：规定时间内到达数量除以本应该到达的数量；

说明我们这种优先级反转的算法是比较稳定的

饥饿：一条消息不能在指定的时间间隔内成功发送

也可以做一个关于负载率和

到达率，到达数量，平均响应时间，低关键级组的响应时间

折线图利用率80和100时，低关键级组输出数量/到达率

普通负载和超负载

最大次数、限制时间、每组N=10，分两组

讲一下CANOE是什么东西、功能

设计思路：查看别人的参考文献，根据时间线整理成为自己的。

先编写内容，把核心内容在完善一些

关键词，消息的激活时间与完成时间

负载率低的情况下，不影响高优先级组在每个周期的正常发送，同时还能够降低低优先级组的响应时间。

**4.18**

待做

参考文献的格式问题

防止产生死锁

提一下分析到达率和响应时间的必要性

想法：一种是设置时间，另一种不设置时间

这个连续是指的周期连续，不是每次都发

高优先级不变PQ

低优先级采用FIFO或者rm

画一个调度时刻顺序条

解释一下连续，这里的连续不是指只有某一报文独自发，而是指每个周期都能发

**任何具有高优先级的报文想要在短时间内高频率的占用总线时，在总线被阻塞时都占有不可推卸的责任。**

**周期加补，就是当在空闲时间没有任何报文发送时，会在发送一个，被反转的报文，**

**加一个补充，如何定义这个发送次数，**

公平性和吞吐量

优先级高且发送频率高的

把图片描述转换为提高了百分之多少

**次数的设置不应该是随机和盲目的，应该最好和饥饿的报文有关联，或者说和低优先报文关联很大，**

**为什么这样选择，是因为这样的话效果很好，考虑的是总的利用率**

**根据饥饿程度，动态的调整固定值**

**饥饿因子乘次数为动态反转次数，取整**

**分组依据是否关键，是否优先级相差很大**

**出来一种新的想法，将高低CAN报文都准备成为反转后的样子，都进入反转队列，确保每个报文发送的时候能够被检测到**

**4.29**

**分两种情况，一种是指考虑最高的优先级，另一种是考虑整个高优先级组，论文可以讨论这两种算法。**

**在不同的总线利用率下分别提高了百分之**

5.6问题

启动了该种联系进行动态的换max以后出现了，在该发送的时间段内，不能够进行正常发送

解决：设计思路必须发生一些变化，把原来的基础代码优化了一下，框架有所修改

可以将该因子进行复杂化，比如考虑整个组对该因子或者更多的因子的影响，进而将公式以及实验进行复杂化，

修改了以后可以完美的兼容我的动态系统，但是这个CANoe与c相比有一个问题，

如果这里的jj = %f 写错为jj = %d，则会连带着后面的tsend\_count 也不能正常显示，

write("jj = %f,tsend\_count = %d", ii, tsend\_count);

**指数？幂函数？或者其他的以及自己创造的函数来反应低优先级组和高优先级组的关系。**

**1.这样选择的目的其实总是选择总线利用率最高的那一个**

**2.这样选择的目的其实总是选择总线利用率最高的一组**

既要能够保护优先级高的发送的次数，又要尽量确保低优先级组能够得到机会发送

提出一种类EDF的算法，其运行过程中优先级不会发生变化，除非当其到达其周期还差一个执行时间时，则会让其进行优先级反转，因为是优先级反转

可以运行在CAN、 CANFD，flexray等总线上面做测试

利用优先级反转来提高总线的安全性，比如虽然发送了但是需要翻译

采用一种时间计时，确保每一个发送的数据都是独一无二的

外文期刊，中文/外文的硕士论文

5.8

能否根据报文的执行时间来动态的安排每次发送的数量，已知报文的执行时间，如果小于1ms的话无所谓，如果大于1ms的话，可以将执行时间相加凑整等….

5.9

不管是高优先级组还是低优先级组都采用这种方法，即高优先级反转，这样的话可以避免存在恶意攻击持续的发送占用总线，但是如果发送的报文有问题该怎么样解决呢， 这时候我感觉应该将任务联系起来，不能再独立的执行，只有联系起来这样的话，哪怕有一个节点被恶意攻击占领了也是不能将报文的作用扩散到最大。可以在报文发送消息确认是否收到的那一刻，进行密码验证，如果能够确认是同一密匙则可。

或者反转的时候尝试发送联系的任务，或者直接增加一个

可以在发送的时候确定其密码的值，然后在接收方对该值进行确认，可以利用时间加某些固定的数字或者动态的数字来考虑。

考虑执行时间，每一个报文的执行时间必须不一样，只有这样才能识别固定的报文。

根据传输的次数来设置动态数据包，达到的效果是一样的，只不过每次传输的数据不同，可以设置一个周期，双方来确定不同的数据包，最好是随机的，这样尽量减少可被发现的规律性。

5.10

安全方向好像大多数都是用的算法来解决的，即python，从头开始学好像有很大的问题，需要找到一个类似方向来研究

不管用不用的到的ID都需要把其列入框架当中，避免被攻入一直发送

但是处理不了，因为当ig发送器发送时在capl里面解决不了。

明天要精读今天这一篇

屏蔽其他报文的方法可以是on message can.\*或者其他，不让发送，或者调优先级

如果数据的字节比较多。可以考虑将其分为两个报文帧发送

Obd 能从汽车中提取can数据，提出以后，可以在类推到CANFD,Flexray

如果受到入侵了应该考虑让其自动停止或者陷入一个自动处理未完成程序的尾巴任务，然后在结束

5.13

如果采用知名算法进行改进加密的话，可以加一个算法优劣的比较。

对称机制

尝试一下虽然建立了数据库，但是不使用数据库id，而是用capl代替功能

下一步工作，测试一下Blowfish代码，测一下capl能否实现接收数据的功能

想法，将一个报文中多个信号，转换为多个报文多个信号

5.15

解决办法

因为初始使用实例的时候，若使用了信号进行操作（幅值或者被幅值），都会产生报文重复发送的问题，为了解决这个问题使用系统变量来代替该信号，活学活用。 Good

High\_message1.word(0) = @Engine::EngineSpeedEntry;

@Engine::EngineSpeedDspMeter = High\_message1.word(0) / 1000.0;

@Engine::jjj = High\_message1.word(0);

5.17

完成blowfish算法移植到capl

下一步实现blowfishi应用到调度当中

5.18

实现了部分应用成功，但是使用数组L[]的过程中，发现出现了问题，但是没有深究，多定义了一个参数来使用

**考虑添加硬件，但是不使用硬件做测试，而是利用硬件加分。**

**考虑一下，设置一个动态的key来进行反复的加密**

**5.22**

**如果采用这种优先级反转的算法在加密算法里面，不仅能够进行对全部的密文加密，还能够尽量的不影响的响应时间。**

**想法：26个字母，打乱顺序，长度任意**

**还有就是需要改进一下。将加密解密配对的时候，在来一个定时器之类的东西**

**5.23**

**待定：昨天留的问题，改变长度的时候，总是默认取最大值，可以考虑找个中间值，做减法**

**解决。因为数组没有清零**

**为什么使用blowfish加密算法，因为该算法运行的速度最快，能够尽可能的满足CAN总线的实时性**

**采用Acsii表**

**可以试试比较一下几种算法或者直接引用**

**块的输入表示的是明文的大小**

**复杂的采用两者结合简单的采用blowfish，当然我们均使用修改版，可以考虑使用md5加密密匙，看看输出的结果，如果一直不变则不使用**

**加一个mdk 5试试，计数器/截断的MAC，计时器加入时间**

**，每隔一定时间换密匙**

**对付dos攻击 ， ID检测，**

**同一ID，确定发送密文是否相同，不同的话则报警**

**加了消息验证以及加密**

**密匙key由计时器/计数器以及rand值共同产生，最好可以循环，尝试一下结合一下最近发送的消息记录**

**5.25**

**可以采用两种加密，一种加密后直接作为验证值，第二种对第一种密文加密后作为报文发送，或者进行两次验证，两次加密后的密文均验证，而且采用第一种加密作为验证值，采用组加密**

**Mini-MAC，一种认证协议，适用于要求短消息大小且无消息延迟的许多车辆系统，包括CAN总线。Mini-MAC显着提高了CAN总线的身份验证强度标准，可防止伪装和重放攻击。可以加一句，同时对dos攻击有一定的防范能力**

该[控制器区域网络](https://www.sciencedirect.com/topics/computer-science/controller-area-network)（CAN）总线是一个简单的，低速总线设计成简单的网络节点。在汽车中通常以500 kbps的速度运行[[9]](https://www.sciencedirect.com/science/article/pii/S2214209616301619" \l "br0090)。 International Organization for Standards

ISO 11989. Technical Report

International Organization for Standards (2015)

**6.7**

**MD5是一种摘要算法又称哈希算法、散列算法，它的计算是单向的，不可逆的，通过digest反推data却非常困难。而且，对原始数据做一个bit的修改，都会导致计算出的摘要完全不同，故不可用来对明文进行加密，但是可以对对称算法中的密匙key进行加密。且原始的可以也是处于动态变化当中。故破解非常困难。MD5严格意义上来讲，不是加密算法，而是摘要算法/散列算法，或者叫做哈希算法。**

**不能对密匙key进行加密，这样没有意义，因为Blowfish的密匙不能经常更换，不然会影响速度，故可以应用在不同的关键级**

**减少Blowfish迭代次数**

**加密算法的目的，在于别人无法成功查看加密后的数据，并且在需要的时候还可以对数据进行解密来重新查看数据。而 MD5 算法是一种哈希算法，哈希算法的设计目标本身就决定了，它在大多数时候都是不可逆的，即你经过哈希算法得出的数据，无法再经过任何算法还原回去。所以，既然不能将数据还原，也就不能称之为可以解密；既然不能解密，那么哈希的过程自然也就不能称作是「加密」了。**

**2.3 md5是唯一的吗？**

**md5作为数据库中的主键可行吗？这就涉及到一个问题，md5值是唯一的吗？答案是，不唯一。也就是一个原始数据，只对应一个md5值；但是一个md5值，可能对应多个原始数据。**

**密文混合以后，**

**对比几种加密算法**

**6.26.2021**

**关于硬件实现的话，目前想到的，加密的报文需要提前都要准备好，和两者的密匙一样。**

**7.4**

**车载CAN总线的通信加密算法研究**

**考虑一下映射**

**提高算法加密强度，第二个是节约时间**

**组密匙**

**计数器可以采用二叉树的几种排序顺序来计数**

**0 0**

**1 2**

**2 6等等**

均方误差或卷积

轮速、发动机速度、车速、里程表、刹车压力、转向角)

7.21

软著待审

7.22.2021

解决CAPL中dll动态链接库的加密数据不准确的问题以及其CAPL中dword和byte中的0和1不是简单一回事，数据不准确重新换了加密的.c文件解决问题

7.24.2021

不同加密算法的先后顺序

7.31.2021

**可以增加明文块的长度，**

窃听攻击。在广播网络中，所有参与节点都会收到所有传输的数据包。因此，对手 ECU 可以读取数据传输以检索信息，这可能会导致未来的攻击。数据传输在 LaaCan 中加密。因此，窃听攻击不能危及数据。

欺骗攻击。欺骗攻击是通过冒充另一个 ECU 来完成的。对手可以通过执行这种攻击来误导车辆子系统。如果网络缺乏真实性和完整性，这些攻击是可能的。LaaCan 确保传输消息的真实性和完整性，以防止欺骗攻击。

重播攻击。在重放攻击中，传输先前记录的有效消息的副本。由于消息是以广播形式交换的，如果不采取安全措施，任何连接到网络的攻击者节点都可以进行重放攻击。[19](https://link.springer.com/chapter/10.1007%2F978-3-030-63095-9_12#CR19)]。当不能确保消息新鲜度时，接收节点无法区分有效消息和重放消息。LaaCan 通过在确保消息新鲜度的密钥流派生过程中使用消息计数器来消除重放攻击。

中间人 (MITM) 攻击。MITM 攻击秘密控制两个合法节点之间的通信。布蒂吉格等人。[[12](https://link.springer.com/chapter/10.1007%2F978-3-030-63095-9_12#CR12)] 通过在仪表盘和车辆模拟器之间安装攻击者节点来演示 MITM 攻击。在 LaaCan 中，攻击者节点无法从加密消息中检索实际信息。此外，可能更改的中继消息无法绕过身份验证过程。因此，LaaCan 可以防止 MITM 攻击。

远程攻击。现代车辆有一个 OBD-II 端口，可提供自我诊断和报告支持。OBD-II 连接到车载网络。吴等人。[[33](https://link.springer.com/chapter/10.1007%2F978-3-030-63095-9_12#CR33)] 提出了将恶意移动应用程序连接到第三方 OBD-II 诊断工具的远程攻击。由于缺乏数据真实性，这些攻击是可能的。LaaCan 提供真实性以防止这些攻击。密钥和加密机制必须与受信任的 OBD-II 加密狗制造商共享，以使它们与提议的架构兼容。如果加密狗易受攻击并且密钥被泄露，则攻击者可能会绕过安全措施。但是，它在某种程度上适用于所有加密方法，因为它们基于密钥进行操作。

**8.1**

**单独检测每一个ID发送的周期或者时间，如果小于该时间发送的则是异常的数据发出警报并不给予正确动作，并找出在该时间一直发送的报文，给予提示。可以找1000ms内出现的次数最多的那个往往就是bug。**

**以及正常的流量监测，流量突然增加并一直维持则发出警报**

**8.8**

**提出一种类似“金字塔”的时间计数器。**

**最底层用来累积发送的次数，**

**往上一层用来作为“秒”，**

**每秒清空一次底层，**

**往上一层作为“分”；**

**最高层作为“时”**

**待做：**

**1.C++中整理DLL，添加AES以及DES加密算法；**

**2.调试加入的算法，实现目前设计的功能；**

**3.继续学习与回顾C++知识**

**8.29.2021**

**考虑一下增加追究消息的来源，可以采用异或的方式来与输出的密文进行，加一个if语句进行判断，如果发送的为IDxx则将相应的识别码增加上去**

**基于频率，但是有一个问题，就是非周期的如何判断**

**9.13.2021**

**增加同步ID，**

**设计成不同的加密格式，比如前面是同步的时间32bit，后面是加密报文32bit**

**Z1= 2x+1**

**Z2 = y -列数;**

**9.20.2021**

**现在的框架是**

**Bf 部分新鲜值+发送ID**

**Md5 全部新鲜值**

**然后用bf和md5异或，在将低四位变成明文的缩密**

**缩密：将原来八字节的数组变为4字节，这四个字节高16位代表1的位置，**

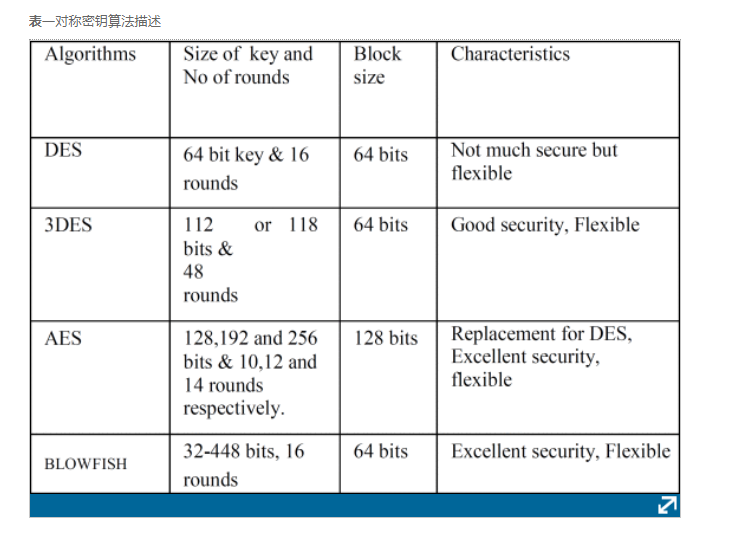
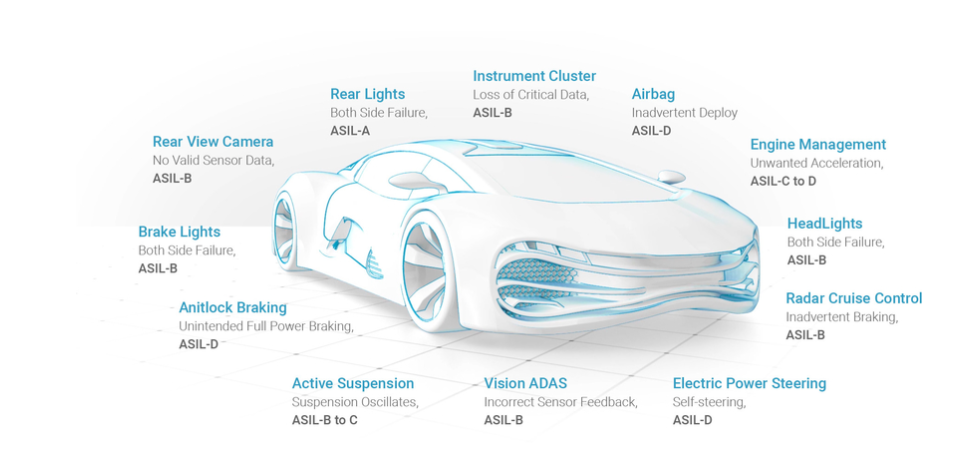
**低十六字节代表速度值在数组中的位置**

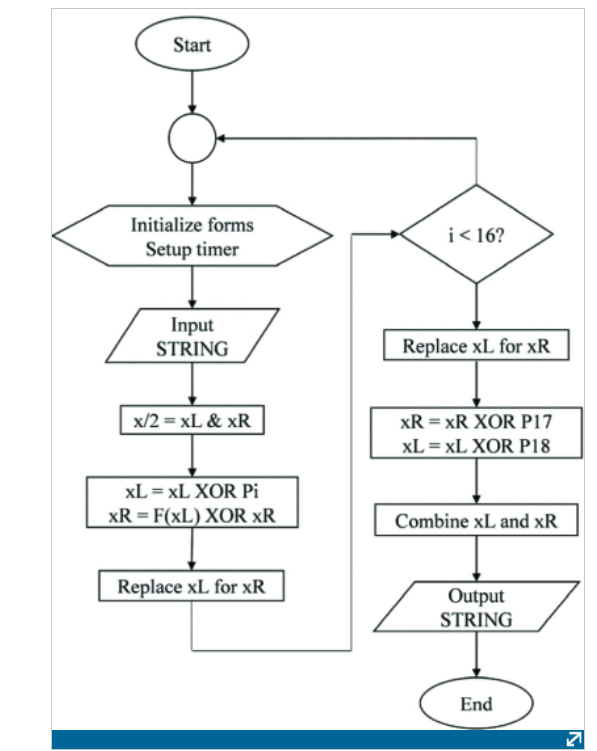
**整理一个缩密的公式**

**然后在与新鲜值进行异或，至此加密完成**

**解密首先需要用新鲜值异或密文，然后 需要同样计算bf和md5，将两者进行异或，然后取高四位与进行操作后的密文比较若一样，则证明该数据可以使用，然后在从缩密中找到对应值即可**

**1.CAN安全，然后最后延伸到CANFD**





**目前两条路走，一个是混合，一个是安全方面**

**综合考虑走安全方面以后应该希望大一些。**

**书363.DLL**