我们的课题是用rust来重写一个嵌入式操作系统，首先我们来看一下我们的项目背景， rust这门语言呢，是Mozilla公司开发的一款新兴语言，于15年发布，最新版本已经更新到了1.25。Mozilla这个公司大家也许不是特别熟悉，但是大家一定非常熟悉火狐浏览器，火狐浏览器就是Mozilla公司开发的产品，你们打开firefox就会显示Mozilla Firefox。 实际上呢，最新版的火狐浏览器已经搭载了用rust编写的浏览器内核，这也是rust的实际应用之一，之后我们还会提到。

然后呢，rust语言有以下几个特点，第一， rust吸收了其他许多语言的优良特性，比如说不需要进行内存管理，所有的内存管理任务都交给计算机处理了。此外rust中没有NULL指针这个概念，这个和c语言是有明显不同的。NULL指针虽然有的时候c语言中用起来挺方便的，但是它会导致严重的安全问题，比如说CVE-2010-4258这个漏洞，就是利用了NULL指针导致的零地址访问异常来提升用户权限，相当于你从普通用户直接就获得root权限了，具体网上中文英文的资料都有。

第二，高安全性。我们知道虽然现在的操作系统内核大部分都是用c语言编写的，但实际上，c语言编写的代码有些地方很容易产生漏洞，比如说c语言不会对数组越界等情况作出处理，但是他编译依旧让你通过，仍然能让你在那边输入输出，仿佛啥都没有发生。但是rust是一门极其讲究精致的语言，他是不会让你轻易编译通过的，c中一些既不会报错也不会报警的问题跑到rust里面很有可能上来就直接给你好几个error。所以rust的编译是很难通过的，但是一编译过基本跑出来也没啥问题。所以有人也说，rust在未来是一门可以取代c和c++的一门良好的语言。

接下来我们再谈谈嵌入式操作系统，嵌入式操作系统呢，它最初是MIT的研究员为了阿波罗计划而研制出的一套导航和操作系统，后来IEEE也就将有控制、监视、辅助操作设备的装置归为嵌入式，而为这种装置服务的操作系统就叫嵌入式操作系统。嵌入式操作系统有很多分类，比如说按照功能分，可以有军工航天、移动设备、工业医疗等分类，按照硬件分类，可以分为……………………，其中嵌入式片上系统，也就是System On Chips，是目前相当热门的嵌入式系统研究方向，在我们现在常用的智能手机和平板电脑中都有着相当广泛的应用。目前从广义上来看，移动设备仍然可以归入嵌入式系统层面，尽管其功能的日渐多样化已经不再符合传统的嵌入式系统的定义了。

然后接下来是我们的立项依据，这张图是盗的2016年全球软件开发大会的，从图上可以看出，C、C++、Rust都具有很好的控制性，因为这三种语言都能让程序员直接对内存进行控制，而JAVA、Ruby还有HASKELL等语言其控制性就较差了。

但是，C、C++的安全性却远不如Rust，这些所谓的原罪我们之前也已经提到过了，这里再说一个心脏出血漏洞，这是一个2014年被披露的一个重大安全漏洞，网上的资料比之前说的那个CVE20104258还要多，简单的来说就是利用了c语言对于边界检查的漏洞来获取用户的账号密码。此外，浏览器也很容易出现安全问题，用c语言写成的内核很容易受到攻击，这也就是为啥mozilla要用rust给firefox重写内核的原因。

所以呢，rust这门语言也就有了相当的重要性了。其突出的三个特点，也就是安全性、高效性、并发性。

首先是安全性，除了之前说过的，rust摒弃了c语言的缺点，它还将一个对象和一个变量绑定，避免了一对多造成的冲突和冒险。还有支持绿色线程也是rust的特点之一。

Rust有所有权这个概念，这就像寄存器只有一个写端口而有若干个读端口，这就保证了写数据的安全性和高效性，不会引发冲突和竞争。

Rust放弃了GC也就是垃圾回收系统，GC虽然是一种比较成熟的系统，但是它也存在着一些明显的问题，比如说内存抖动，内存抖动呢这个老师上课刚讲了，我这里就不再多嘴了，而采用的这个新架构能够很好地解决抖动的问题。

最后，我们再来谈一下freeRTOS，其他有小组也做的是这方面的东西，这里我就说的简短一点，freeRTOS作为一个轻量级的操作系统，具有可移植性好，能在多平台上使用，最重要的它是开源的，我们可以免费阅读其代码并基于rust的特点进行修改和移植，能减少我们相当多的工作量。

FreeRTOS在中国也有着相当广阔的前景，随着智能手机等移动设备的不断普及，再加上中国巨大的人口基数，嵌入式系统的普及化几乎是肯定的事情了，这也正是我们选择这个项目的原因之一。

那么接下来就到了相关工作这部分了。由于rust是一门比较新的语言，所以基于rust并且已经完成的项目并不是特别多，目前已经完成的，除了Mozilla本公司用rust重写的firefox新内核外，还有TiKV一个分布式数据库，具有相当高的保密性和安全性。此外，在GitHub上也有名为Redox的操作系统，这是一个纯粹基于rust语言来开发的操作系统，他的目标就是为用户提供一个开源的、安全的Unix微内核操作系统，和我们这个项目也有类似之处。

而rust作为新生的一门优秀语言，也越来越为IT界所接受，从2017年rust官方网站的调查报告显示，人们对于rust的兴趣与日俱增

而rust的普及也体现在国外的高等教育中，University of Stanford的计算机专业今年春季学期就开设了一门操作系统专业课cs140e，这个e就是experimental的意思，这是一门试验性质的革新课程，和传统的操作系统课程相比，未使用任何虚拟机或者模拟器，也不适用c语言等传统的系统编程语言，而是采用了树莓派加rust的模式来构造嵌入式操作系统，实现虚拟内存、文件系统等功能。Emmmm树莓派这个，我们没钱就不用啦，不过老师也为我们组提供了一个开发板，我们接下来的工作将基于这块开发板来进行。

这个开发板是基于arm架构的，搭载了蓝牙5，而且仅从目前来看，SoC，也就是system on chips，是嵌入式系统发展的一个重要方向，其高度集成化的特点非常适合于物联网技术的发展，这也是我们研究的一个重要方向之一，之后我们还会说到这一点。

然后就是可行性分析部分了，总体看来我们这个项目的可行性还是相当高的，毕竟无论是业界还是教育界都有向这方面发展的趋势。我们唯一的难点可能就是工程量和时间上的矛盾了，下面我们先来看一下理论上的依据。

首先我们要探究的是，rust对嵌入式操作系统究竟能支持到什么程度，也就是说，用rust来编写嵌入式操作系统是否可行，从之前的介绍中，我们可以看出，rust和c一样，是一门非常偏向底层的语言，而且从斯坦福大学以rust语言为平台来进行操作系统的教学来看，rust对嵌入式操作系统的支持是完全没问题的。

此外，我们的开发板基于arm架构，而rust中也有asm！宏可以调用，我们甚至可以在rust中直接嵌入汇编代码来直接调用对应的寄存器，且能支持arm指令集体系。而且最重要的是，rust具有很多外部函数接口，这也使得rust和c的混合编程成为了可能，目前rust的用户群中，也有相当一部分企业是选择用rust和其他语言进行混合编程的，而且rust这个语言，尽管刚发布时有不少不稳定的现象，但是到了1.6版本，这个问题就得到了很好的改善。rust的核心部件之一的libcore的稳定性得到了很好的改善（libcore是rust standard library中的一部分，把语言和函数库对接到了一起，是rust中相当重要的成分之一。）

然后是我们的技术依据，我们想过好几种策略，第一种是完全将现有的c语言体系推倒重来，自己写一个rust版的RTOS，这个优点是能充分利用rust的语言特性，但是工作量……要是我们真能写出来早就不呆在这里了。然后第二种策略是根据已有的freeRTOS中的函数模块，来实现对应的rust的版本，这个的可行性就比刚才那个高很多了，但是也存在着相当大的工作量，但是若是我们充分阅读c代码之后，便可以直接在c的基础上改写成rust代码，这样子一方面可以减少工作量，另一方面还能增加对rust相较于c的优越性的理解。所以我们的想法是利用rust来重写一个嵌入式实时操作系统，至于为啥不选这两个的理由，一是工作量太大，二是其无法彻底发挥出开发板特性。

而基于c语言的技术实现论证，我们总结了以下几点：第一，重写内核的工作量是难以想象的，需要对内核有着相当充分的理解的基础上，还要能重写成rust语言，这对我们来说是亚历山大，而且，c语言和rust有着良好的兼容性，所以我们可以对一些核心函数用rust进行修改，而一些本身安全性就很高的部分就可以直接调用接口实现。这也正是利用了rust和c的编程相似性和兼容性的特点。

然后最后就是一个中期汇报的总结了（终于tm要完了）。从科研的三部曲来看，目前RTOS这方面的需求就是物联网的不断发展，导致嵌入式系统的需求量不断增加，未来可以说是人手一部智能手机，此外，家里的微波炉、冰箱、乃至台灯、充电器等等，都会安装嵌入式操作系统，而此时如果被黑掉，那么势必会对人们的日常生活和经济方面都造成相当严重的损失，所以我们势必要增加嵌入式系统的安全性。

但是c和c++的天生的弱点导致了嵌入式系统的安全性不足问题无法得到有效解决，所以我们必须转而使用更加强有力的开发工具。

而rust这门语言正是我们最大的创新点所在。Rust作为一门新生的语言，其安全性、高效性也得到了业界的认可，国外的高校也将之用于教学研究，也有人说，rust在未来是最有希望取代c和c++的语言，因此，我们学习rust并利用它进行开发，以达到以下目的：

第一个，最有希望实现的，编写一个轻量版的操作系统，增加其安全性、并发性、高效性。

第二个，鉴于老师也提供给了我们开发板，我们希望能充分利用其蓝牙功能来增加我们编写的操作系统的实用性和可行性，使其适应SoC的大发展方向，在开发板上发挥出其应有的功能。

至于最后一个目标啊……emmmm愿望是远大而美好的，我们都想融入时代潮流，做新时代的弄潮儿，不过……慢慢来，慢慢来啊，一步一个脚印先走下去吧。

然后这个是我们的参考资料，rust这方面的研究大都是比较新的，而且数量较少，所以相关信息都是从官网和一些比较新的论文上上找来的。

嗯，大概就是这样，谢谢大家！