实验二　openEuler 进程调度

一、　实验说明

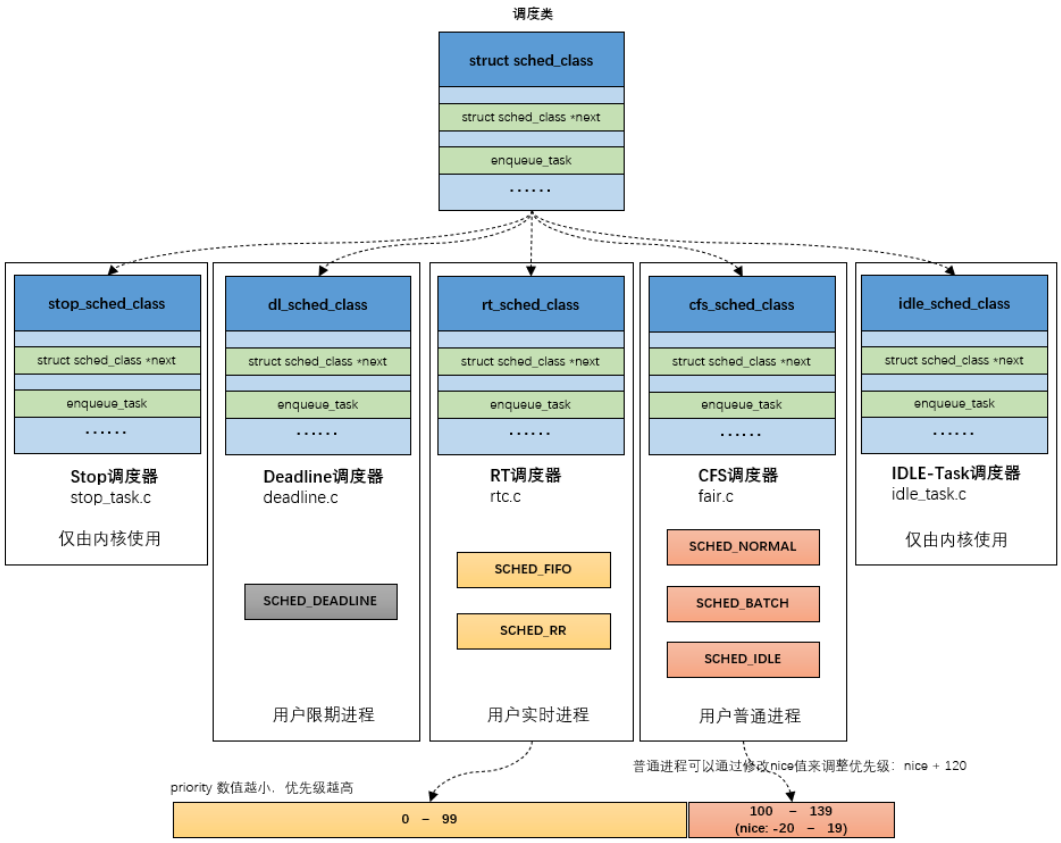
实验二与实验三都包括两部分：必做+选做，每部分各占一半分值，所有同学只需要完整一个完整的实验（必做+选做）或两个实验的必做部分，总分即达到要求。即完成总分有三种基本选择：

1、 实验二必做+实验二选做

2、 实验三必做+实验三选做

3、 实验二必做+实验三必做

二、　实验背景（以 4.19.90-2209.5.0 为例）



Linux 使用了基于优先级的时间片轮转法等调度算法进行进程调度。它使用各种策略来确定进程的优先级，在每个 task_struct （kernel-4.19.90-

2209.5.0/include/linux/sched.h）结构中都有以下信息：

(1) 策略（policy）

Linux 中有两类进程，普通进程和实时进程。

（2）优先级（priority）

（3）实时优先级（rt_priority）

Linux 内核抽象了一个调度类 struct sched_class （kernel-4.19.90-2209.5.0/kernel/sched/sched.h），这是一种典型的面向对象的设计思想，将共性的特征抽象出来封装成类，在实例化各个调度器的时候，可以根据具体的调度算法来实现。这种方式做到了高内聚低耦合，同时又很容易扩展新的调度器。

Linux 的调度函数是 schedule（kernel-4.19.90-2209.5.0/kernel/sched/core.c）。在 Linux 中，绝大多数进程采用的是完全公平调度算法 CFS，引入了红黑树结构来存放运行队列上的每个进程。

三、 实验目的

1、 以 openEuler 为例熟悉 Linux 的进程调度代码；

2、 掌握 CFS 算法流程和 schedule（）调度函数流程；

四、 实验要求

1、 必做部分

结合自己之前替换的 openEuler 内核版本源代码，（以 4.19.90-2209.5.0 为例），CFS 的源代码主要在 kernel-4.19.90-2209.5.0/kernel/sched/fair.c 文件中，调度函数 schedule（）在 kernel-4.19.90-2209.5.0/kernel/sched/core.c 中。查阅相关资料并结合核心代码部分，写一份报告，包括 CFS 相关代码所实现的核心算法流程图、红黑树在进程

调度中的作用、重要的数据结构、schedule（）函数执行流程图等。

2、 选做部分

修改源代码并重新编译内核，实现功能：记录进程调度过程中切换前后的进程号。报告中需写清修改的代码部分并进行解释，修改过的代码文件需写好注释，实验结果等。

五、 提交内容

最终提交的内容（根据自己的选题情况）可能包括：

1、 必做部分的报告（重点是对调度过程的理解，关系图、流程图）；

2、 选做部分的报告，选做部分修改的代码。

六、 提交时间

5 月 21 日（十四周周日）23:59 之前将文件提交至 CANVAS。

注：第二次实验与第三次实验将合并在第三次实验中提交，截止时间相同。提交时写清自己的选题情况。