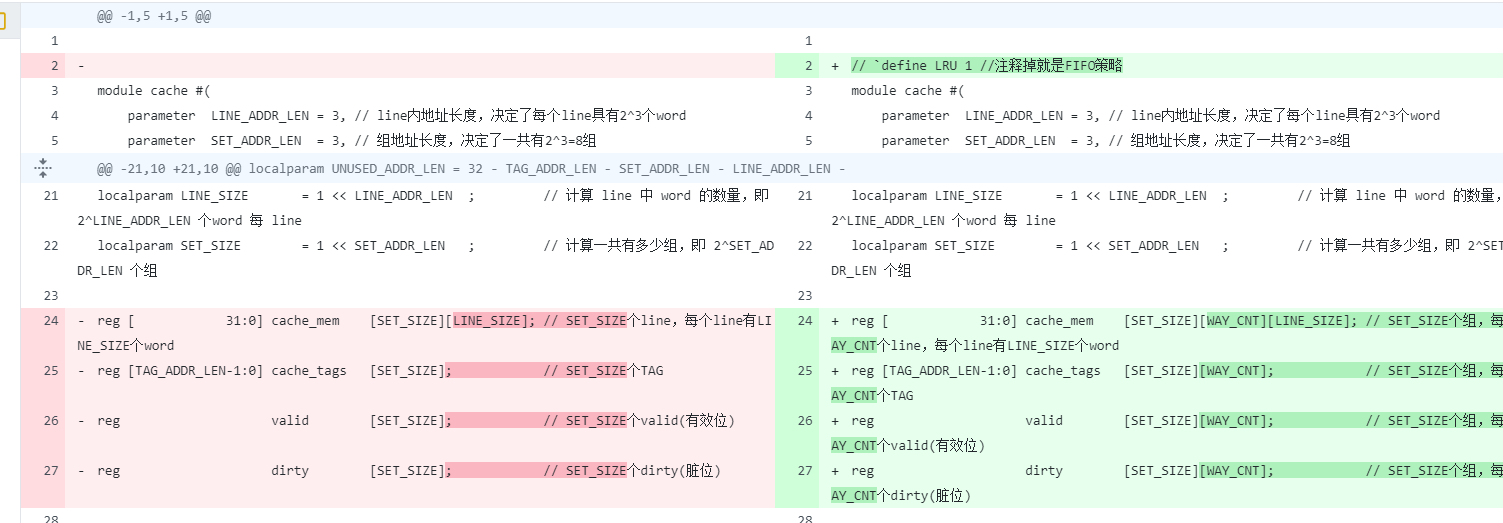
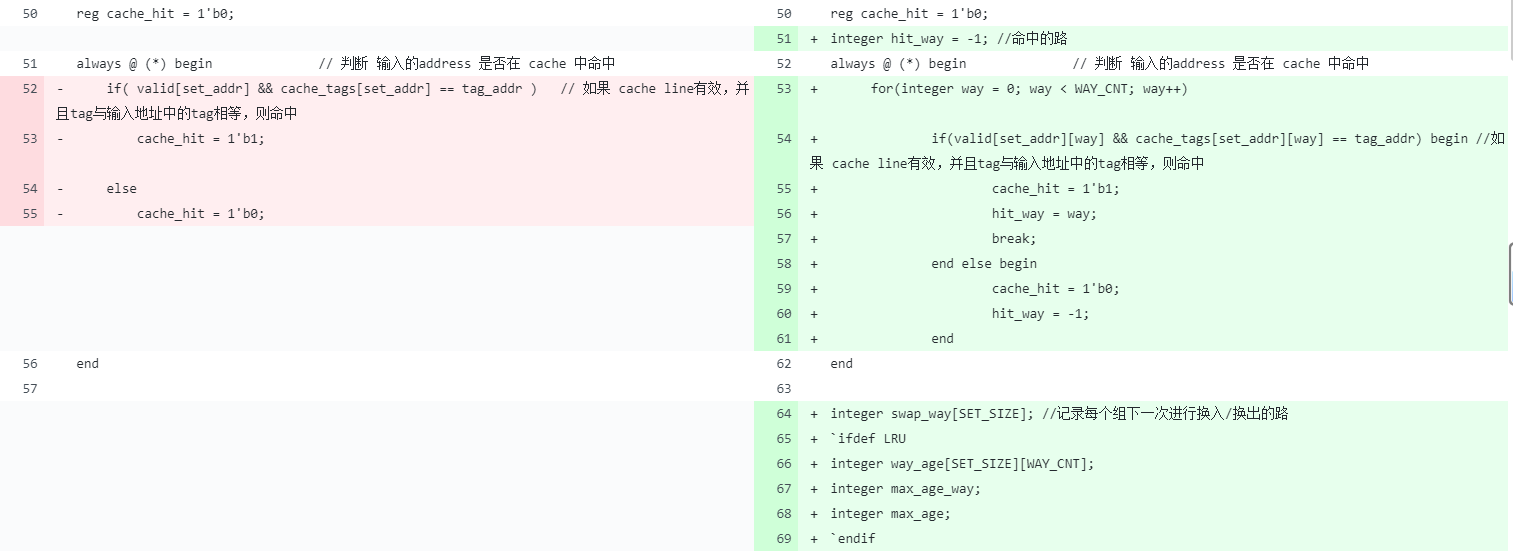
在理解原cache代码的基础上，将其修改为组相联的cache，并实现LRU和FIFO两种调度策略：（为方便起见，使用github的diff工具进行代码修改说明）

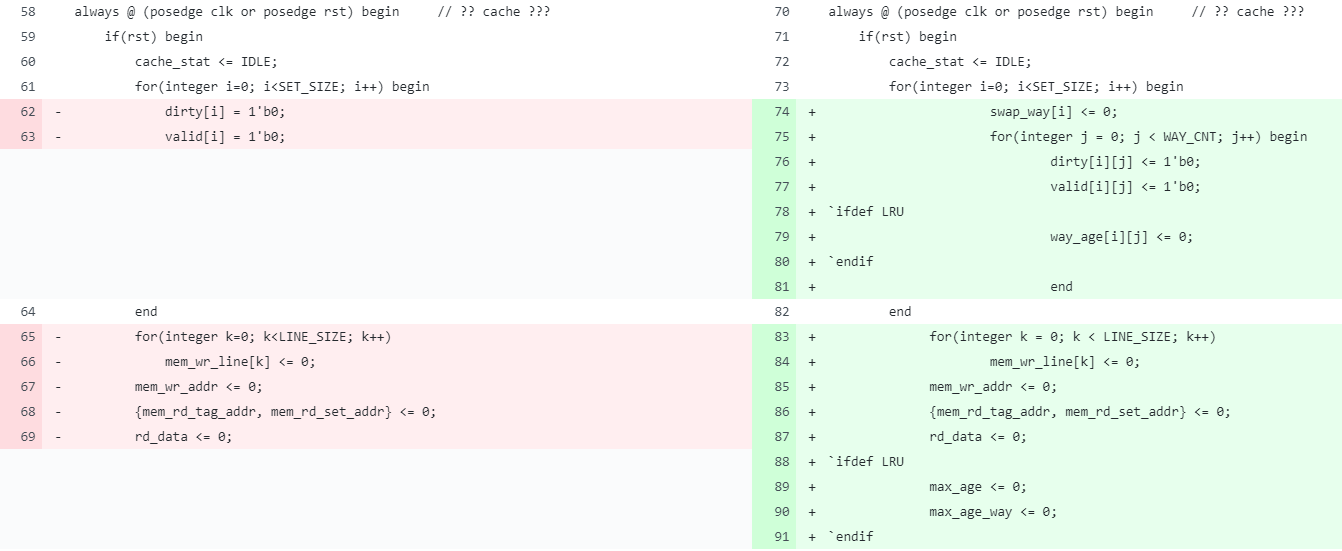
1. 首先，采用宏定义切换LRU和FIFO：



1. 其次，在reg中增加数组的维度（增加了[WAY\_CNT]，图如上）
2. 然后修改cache命中部分，增加变量记录命中的是哪一路，为实现置换策略，增加数组记录下一次换入换出的路（LRU需要额外的存储空间，记录每个路的年龄和最大年龄的路）：



1. 修改begin逻辑，初始化我们增加的变量：

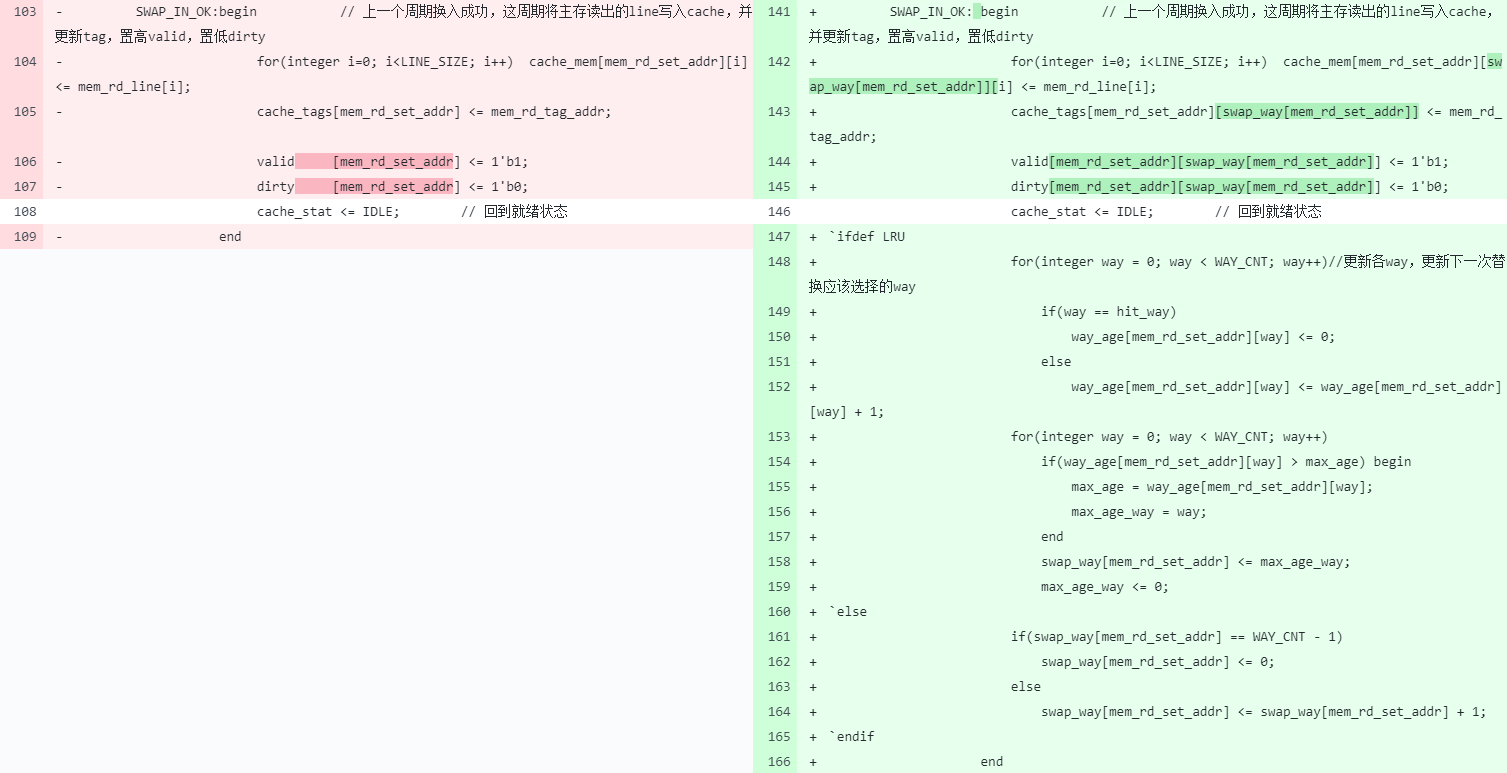


1. 修改idle逻辑，主要是各数组增加维度后要做的修改，以及处理LRU的年龄更新逻辑（LRU年龄就是未命中就加1，命中了就归0，然后再找出max\_age）：





1. 接下来的很多中间状态并不需要处理，直接修改SWAP\_IN\_OK中的逻辑即可，这里由于LRU和FIFO的逻辑不一致（FIFO只需要数组值在0和WAY\_CNT-1之间就行了），因此采用了ifelse的写法：



1. 其余逻辑不需要做修改，可以直接沿用。