**寄存器安排方案**

这所谓的寄存器安排方案主要是指，我应该用什么样的方式去描述某个架构下的物理寄存器（如果有需要，可能特指x86架构下，因为mcc目前是x86下的）。

寄存器分配需要额外考虑两件事情：1、寄存器预着色2、寄存器别名问题。

对于寄存器预着色问题，在逻辑寄存器的实现中，可以额外的指定该变量存放的寄存器，从而在逻辑寄存器映射到物理寄存器的过程中，指定该寄存器。而相应的，在该变量分配过程中，可以指定一系列寄存器，表示在这些寄存器范围内，都可以进行选择。

对于寄存器别名问题，比如ax和al、ah冲突的问题。正是我在这里进行设计描述的目的。

首先，将所有的寄存器，前后相接，看成一个连续的空间，而某个寄存器，则指向该空间，并且，给出指向该空间的长度，来表示这个寄存器的长度，比如ax和al，ah就可以指向不同的偏移和size。

|  |  |
| --- | --- |
| Ax寄存器（off：0，size：2byte） | |
| Ah寄存器（off：0，size：1byte） | Al寄存器（off：0，size：1byte） |

从而用这种方式解决别名问题。

显然，这和union的情况是类似的，因此，可以直接按照这种方式去实现。统一封装成一个register file的struct结构体即可。而如果使用了某个寄存器，则在该数值处，置位为1。

那么，要如何在逻辑寄存器中描述某条指令可以使用的寄存器呢？

我认为，给出对应的off和size，并且所有可能的情况拉成一个链，这是在指令描述中存放的。而且，源操作数和目的操作数都需要考虑进去。

而当某个指令进行分配的时候，则从其可预选的一些寄存器中，进行查看，探测其是否全部空白，如果是，则进行分配，否则，弹出（算是不考虑spill下的图着色吧。）

当然，这里面同样需要考虑一些特殊情况，比如说，x86的flag寄存器，尽管他改了吧，但是又不影响，具体情况需要具体分析。

**寄存器分配**

这里主要描述寄存器分配的一些常用算法和我个人对此的理解

1. 图着色
2. 线性扫描
3. PBQP
4. SSA

**寄存器spill的选择**

寄存器如果发生了冲突之后，选择哪个寄存器用于弹出并压栈，然后给他空出一个空间，是非常重要的一个事情。