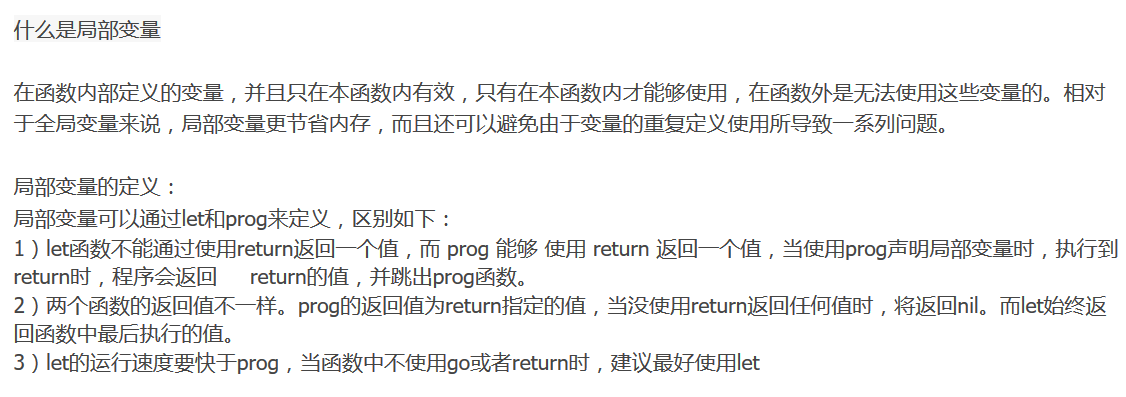
论坛：

<http://www.allegro-skill.com/forum.php?mod=forumdisplay&fid=65>



# 三种使用skill的方法

1. 在命令行键入命令
2. 使用快捷键
3. 使用菜单

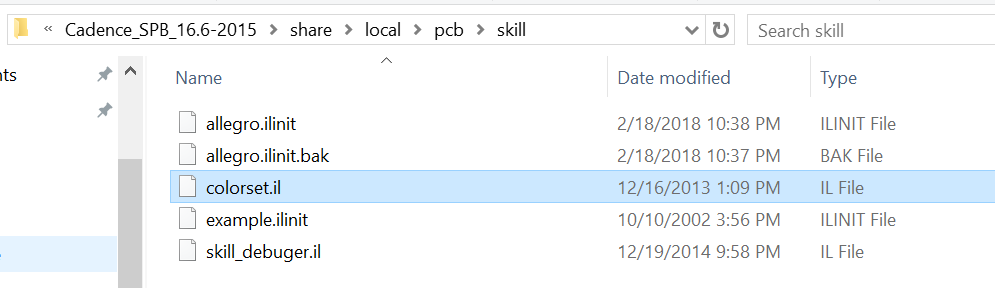
## 1，举例说明：在命令行键入命令

比如有一个skill文件：colorset.il

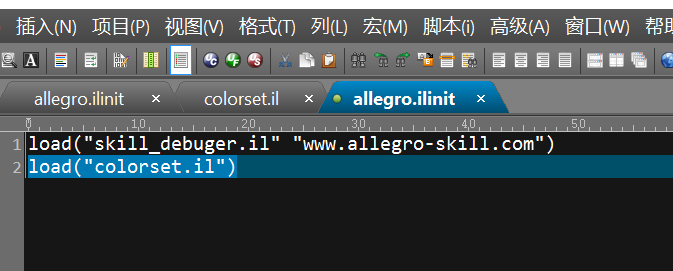
他的的功能是设置颜色。经skill debugger验证可以正常使用，但是不能每次都用skill debugger加载一遍啊。

编写完这个文件怎样才能把他应用到实际工作中去呢？

### ●首先将skill文件放到指定目录下：



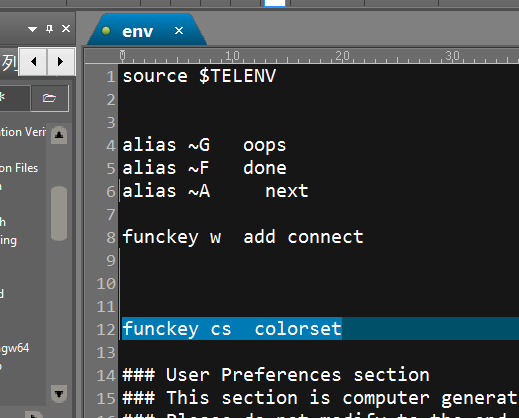
### ●然后修改原来的或者新建的allegro.ilinit文件。



### ●重启allegro，在命令窗口中输入下图所示命令，实现快速颜色设置：



## 2，举例说明：使用快捷键设置

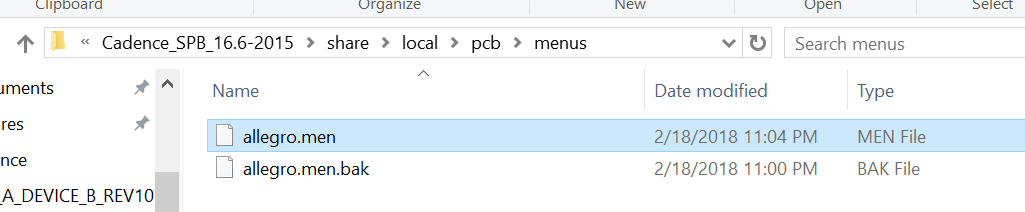
在1的基础上，在环境变量中添加快捷键设置：

然后重启allegro，在当前窗口为PCB编辑界面的情况下按键盘上的C，S键即可完成颜色设置。（不过原则上来说频繁的操作才有必要设置快捷键，像这种开始画板前的颜色设置也就做一次，用菜单就够了）

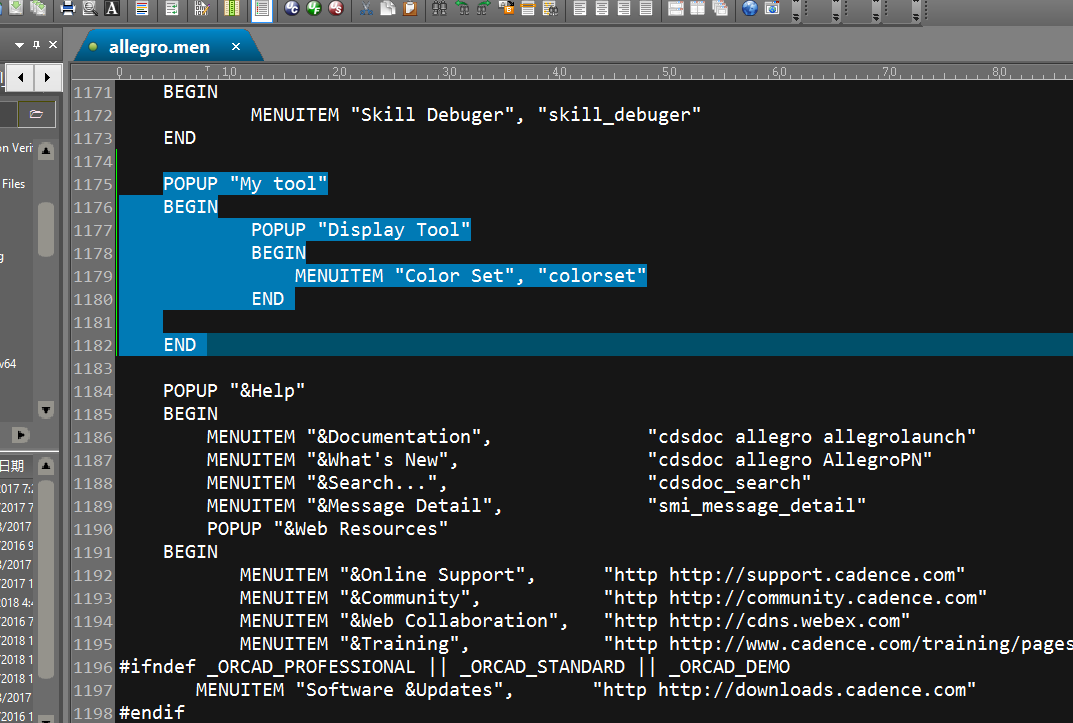
## 3，举例说明：使用菜单

还是在1的基础上。

●操作该目录下的menu文件，如果该目录下没有这个文件就搜到这个文件然后复制过来

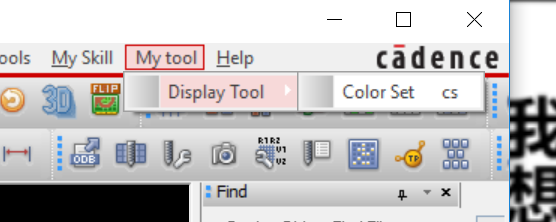


在文件接近最后的部分添加代码：



●然后重启allegro

工具栏中就会出现：



## 小结

这个颜色设置的功能最终应该用工具栏的方式实现，而且还可以进一步修改skill文件，使得点击COLOR SET之后弹出一个窗口，显示常用层和线的默认颜色，在用户点确定之后就可以完成颜色设置，在点确定之前可以对颜色进行调整。这个功能以后再开发吧。

# 调试自己编写的skill文件的方法

很显然，按照上面的步骤就可以：

1. 将编写好的skill文件放在指定目录下
2. 修改allegro.ilinit文件加载skill文件
3. 重启allegro，就可以在命令行输入自定义的命令了。

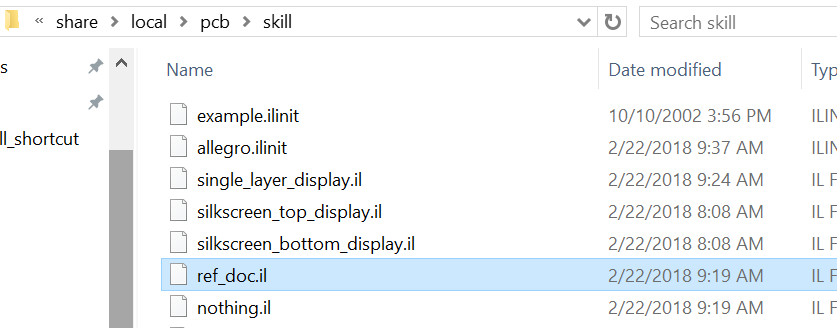
但是这个方法比较不方便，尤其是需要重启allegro。

## skill load("allegro.ilinit")命令

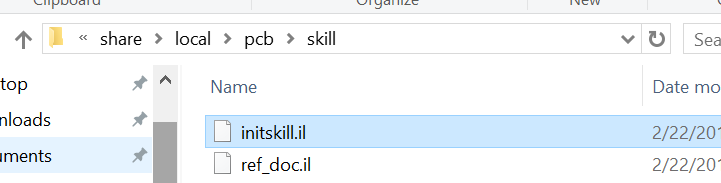
其实用skill load("allegro.ilinit")命令就可以解决这个问题了，他会重新加载skill文件而无需重启allegro。

下面的几部是为了做一种隔离，将调试文件和真正的功能文件区分开：

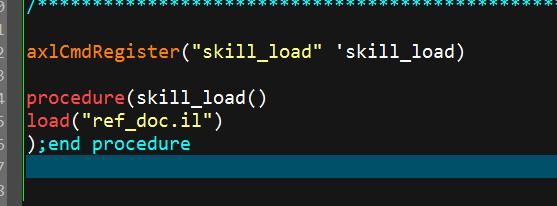
●首先还是将编写好的skill文件放在指定目录下



●然后在相同目录下新建一个skill文件用于加载用于调试的skill文件：initskill.il



主要代码如下：



●然后在allegro.ilinit文件中添加代码：load ("initskill.il")

●为了进一步加快速度，在环境变量中设置快捷键：funckey skl skill\_load

●重启allegro：用快捷键skl 就可以加载rec\_doc.il

●以后，只要修改initskill.il中的load（”xxx.il”）部分，就可以用命令skill load("allegro.ilinit")代替重启操作，然后用skl加载调试文件。

如果不加载新的调试文件，只是修改了xxx.il的内容，只需用skl重新加载调试文件即可。

## 转化与划归

上面发现一个规律：

### 现象：

在不重启allegro的情况下：

执行命令A，

修改命令A所对应的函数

修改命令A所对应的函数中调用的其他文件的内容

执行命令A，

对命令A所对应的函数的修改无效。

但是命令A所对应的函数中调用的其他文件的修改却可以生效。

### 规律：

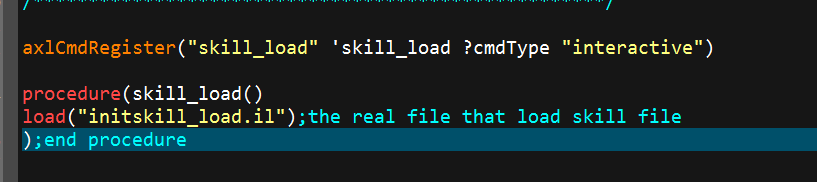
Allegro启动后注册的命令就固定了，修改也无效除非修改后在重新加载命令，但是命令所调用的其他资源修改后可以立即生效。

由此可以用命令A调用命令B的文件的方式让命令B的修改即刻生效。

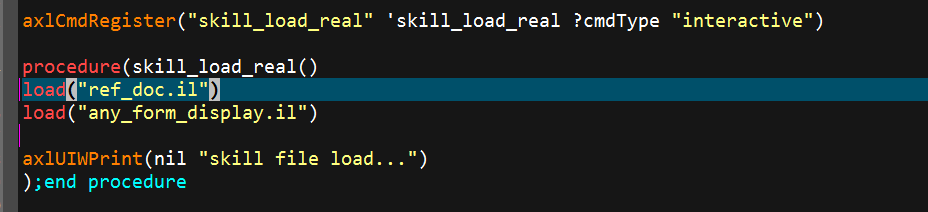
具体操作的时候也要先执行一次命令A，再执行一次命令B。

### 具体操作：

●在环境变量目录下建立initskill.il文件



●在环境变量目录下建立initskill\_load.il文件



●然后在allegro.ilinit文件中添加代码：load ("initskill.il")

●重启allegro，依次在命令窗口输入skill\_load, skill\_load\_real即可加载用于测试的skill文件：ref\_doc.il , any\_form\_display.il

●修改原来的测试文件只要用skill\_load\_real就可以生效。

●在initskill\_load.il中load新的测试文件需要用skill\_load, skill\_load\_real才可以生效

●在initskill\_load.il中删除已经加载的测试文件恐怕要重启allegro

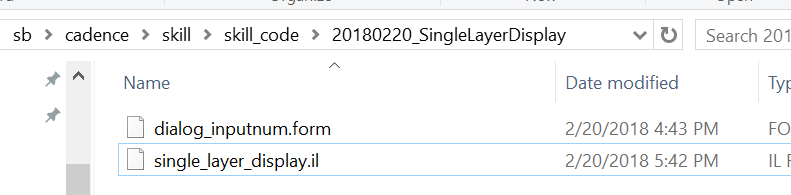
●在环境变量中给skill\_load, skill\_load\_real设置快捷键会更方便

# 建立自己的菜单栏和快捷键

## 一，显示设置

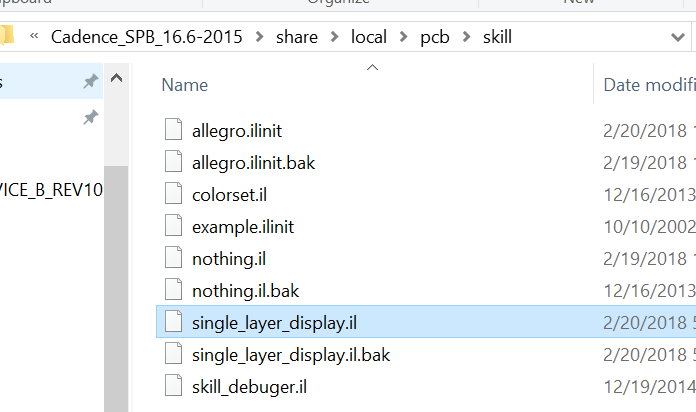
### 1，单层显示和多层显示

单层显示主要靠两个文件：

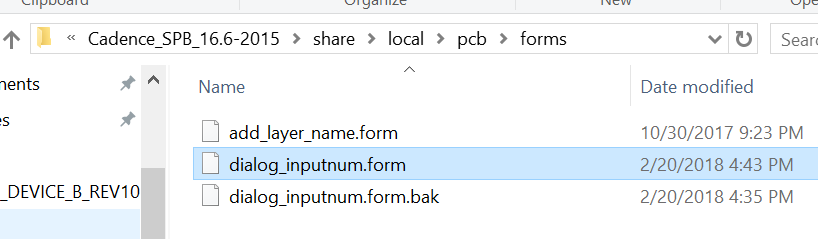


让这两个文件发挥作用需做如下操作：

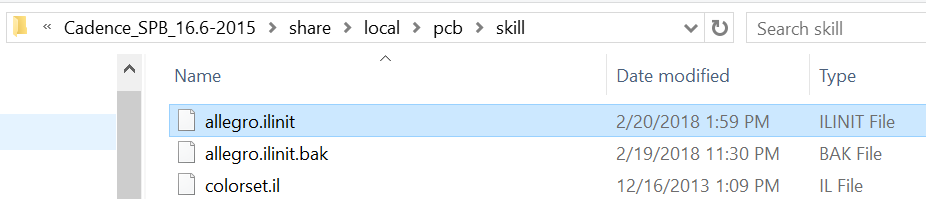
#### ●将.il文件放到指定位置

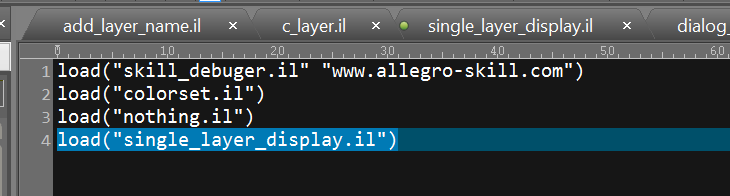


#### ●将.form文件放到指定位置

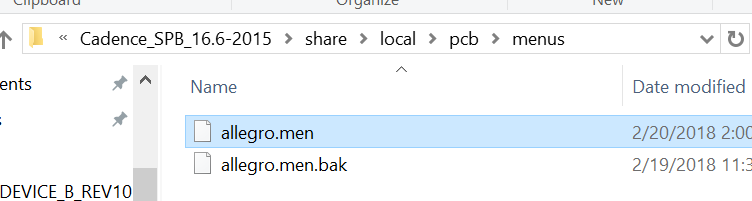


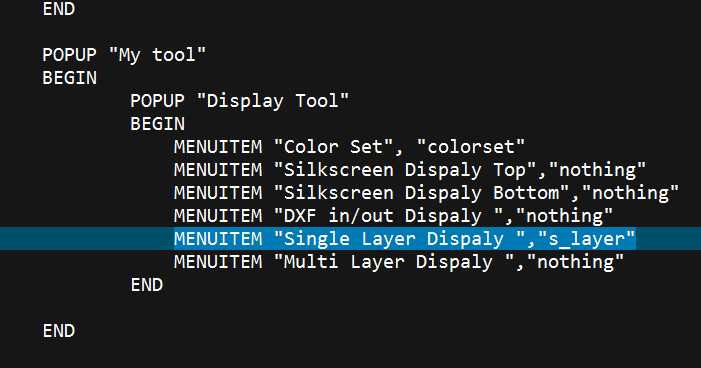
#### ●修改allegro.ilinit文件





#### ●修改allegro.menu文件

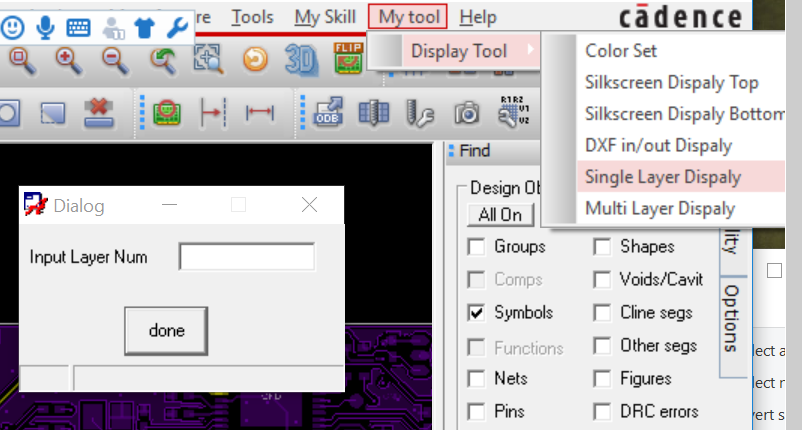




其中s\_layer对应skill代码中的命令注册函数的第一个参数，而不是后面的执行的函数'single\_layer\_display：

axlCmdRegister("s\_layer" 'single\_layer\_display ?cmdType "interactive")

#### 效果：



也可以用命令行+快捷键：

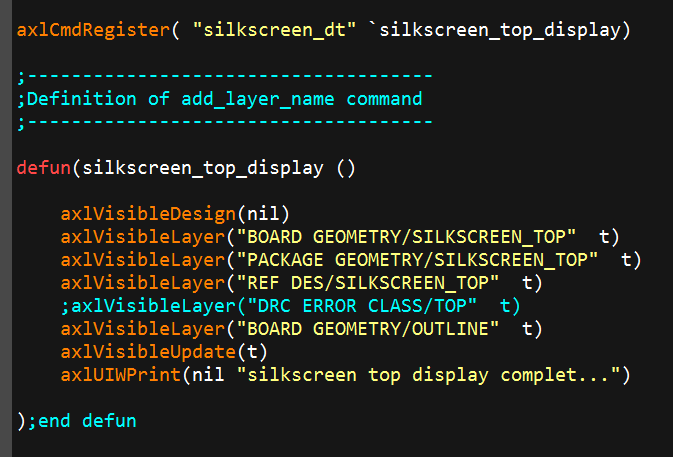
在环境变量中加入：alias sld s\_layer

然后在命令行中输入：

sld + 层编号

多层显示的代码与单层显示几乎相同，只是少了最开始取消所有层显示的代码。

### 2，顶层丝印和底层丝印的显示



### 3，DXF导入和导出的图层显示

和前面的操作基本相同，DXF\_display.il中运用了case语句，这样就可以处理default的情况了。

case(mul

(0

axlVisibleDesign(nil)

axlVisibleLayer("BOARD GEOMETRY/SOLDERMASK\_TOP" t)

axlVisibleLayer("PACKAGE GEOMETRY/PLACE\_BOUND\_TOP" t)

axlVisibleLayer("PIN/TOP" t)

axlVisibleLayer("BOARD GEOMETRY/OUTLINE" t)

axlVisibleUpdate(t)

)

(1

)

(t

axlUIWPrint(nil "Error,Wrong layer argument! ~~~~(>\_<)~~~~ ")

)

);end case

#### 自定义指令注意事项：

不要与系统自带的指令重合。

●最初用DXF\_display作为command，结果导致alegro自带的导入DXF功能失效。

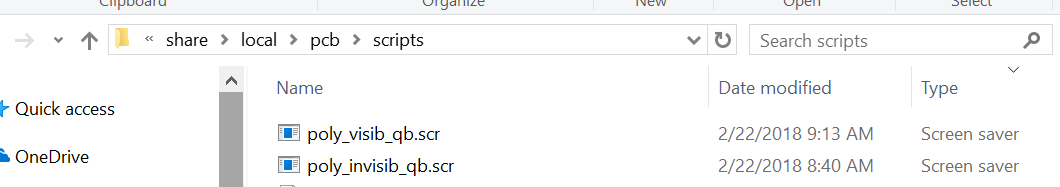
●还有我用alias dxf DXF\_display做快捷键，也导致相同的问题。后来改成

alias IOdxf DiXiF\_display\_qb才解决问题。

●另外在解决这个问题的过程中，我用经过加密的skill\_debuger.il加载过DXF\_display.il，导致无论如何无法恢复DXF导入功能，最后不得不重装cadence 。天知道那个加密的文件做了什么好事！！！

### 铜皮的显示和隐藏（不影响走线）

用录制脚本 获得.scr文件，然后将文件放入下图所示路径。



然后修改.men文件：

MENUITEM "EtchShape Visible","replay poly\_invisib\_qb.scr"

MENUITEM "EtchShape Invisible ","replay poly\_visib\_qb.scr"

也可以修改环境变量自定义快捷键：

funckey siv replay poly\_invisib\_qb.scr

funckey sv replay poly\_visib\_qb.scr

#### 录制脚本注意事项

录制脚本的过程中有一点需要注意：最好是关闭allegro再打开，然后不做其他操作就开始录制脚本。这样录制出来的脚本是最完整的。

[老吴skill出gerber](http://www.mr-wu.cn/cadence-allegro-skill-create-film-record-api/)

[SKILL刷新铜皮](http://blog.csdn.net/wu20093346/article/details/12707505)