GRASS 开发者手册

1 make 选项说明

1.1 Synopsis:

make [-j*n*] [DEBUG=*d*] [GUI2D=*g*] [PARALLEL=*p*] [clean] [distclean]

1. -j*n*
   -j 使编译过程并行化; *n* 指定了并行编译的进程数. 这是 make 程序自带的选项
2. GUI2D=*g*
   g=[0|1]. GUI2D=0 为默认值,编译生成的是 3 维 GUI; GUI2D=1 则会生成 2 维 GUI.
3. PARALLEL=*p*
   p=[0|1]. PARALLEL=0 为默认值, 编译生成的程序只能用单线程方式运行; PARALLEL=1 将在编译过程中加入 OpenMP 并行库, 编译生成的程序 (部分代码) 以多线程方式运行.
4. clean
   清除编译过程中产生的所有中间文件
5. distclean
   清除编译过程中产生的所有中间文件以及最终生成的程序

2 GRASS 运行语法

编译生成的程序名称与当前编译的平台和编译的选项相关, 具有如下形式:

grass-***GUI_TYPE*-*PLATFORM*-*release_type*-*thread_type***

其中

- ***GUI_TYPE***=[GUI2D|GUI3D]
- ***PLATFORM***=[Linux|CYGWIN|MINGW32]
- ***release_type***=[debug|release]
- ***thread_type***=[Serial|Parallel]

例 1, 在 cygwin 中用下面的命令编译 GRASS:

$ make -j2 GUI2D=1

生成的程序名为 grass-GUI2D-CYGWIN-release-Serial.

例 2, 在 Linux 下 (Redhat, SuSe, 或 Ubuntu 等发行版本) 用下面的命令编译 GRASS:

$ make -j2 DEBUG=1 PARALLEL=1

生成的程序名为 grass-GUI3D-Linux-debug-Parallel.

为叙述方便, 以后均用 grass-GUI2D 和 grass-GUI3D 来表示各种编译版本的 GRASS.

2.1 带 2 维 GUI 的 GRASS

带 2 维 GUI 的 GRASS 只接收一个参数 (网表文件名), 运行方式和 HSPICE 类似,

$ **grass-GUI2D** *spice_netlist*

其他操作都在 GTK+界面上完成.

2.2 带 3 维 GUI 的 GRASS

带 3 维 GUI 的 GRASS 运行方式如下:

$ **grass-GUI3D** *spice_netlist* **-c** *configure_file*

其中 *configure_file* 的格式比较复杂, 下面是这类文件的一个模版:

```
Parameter1 alias1
Lower_bound1 Upper_bound1
Number_of_linear_samples1

Parameter2 alias2
Lower_bound2 Upper_bound2
Number_of_linear_samples2

Threshold_on_magnitude_of_tf_for_evaluating_phase_margin
Threshold_on_magnitude_of_tf_for_evaluating_band_width

Start_freq End_freq
Number_of_logarithmic_samples

Standard_variation_of_2D_normal_distribution_refering_to_parameter1
Standard_variation_of_2D_normal_distribution_refering_to_parameter2
corrcoef
```

例如, 网表文件中的两个器件 C1 和 R1 是待分析参数, *configure_file* 的内容如下:

```
C1 C
1e-12 1e-10
60

R1 R
20 2000
70
```

0
-3

1 1e7
200

3.67e-12
73.3
0.0

这个设置文件对 3 维符号仿真进行了如下设置 (设置文件中的关键参数均用黑体标出):

- 3 维曲面上的两个变量轴 "**C**" 和 "**R**" 分别代表器件 **C1** 和 **R1** 的取值.
- **C1** 的取值范围是[**1e-12**, **1e-10**], 单位 F.
- **C1** 在取值范围内线性 (等距) 取 **60** 点, 用于绘制空间曲面.
- **R1** 的取值范围是[**20**, **2000**], 单位 Ω.
- **R1** 在取值范围内线性 (等距) 取 **70** 点, 用于绘制空间曲面.
- Phase Margin (PM, 单位 Degree) 按如下方式计算:
  找到使$|H(j2\pi f)|=$**0**dB 的最小频率 f_{PM}, 则 PM=180 - arg $H(j2\pi f_{PM})$.
- Band Width (BW, 单位 Hz) 按如下方式计算:
  找到使$|H(j2\pi f)|=$**-3**dB 的最小频率 f_{BW}, 则 BW=f_{BW}.
- 各种频域指标的计算局限在[**1**, **1e7**]频率范围内, 单位 Hz.
- 在指定频率范围内按对数尺度 (log10) 采点, 采点数为 **200**.
- 寻找设计中心的过程中用到的 2 维正态分布的标准差分别为 σ_{C1}=**3.67e**$^{-12}$ 和 σ_{R1}=**3.67e**$^{-12}$, 相关系数 ρ=**0.0**, 两个均值由用户从等高线 GUI 窗口(contour window)中选择.

3 维窗口中有很多热键可以用于控制约束曲线等辅助的显示和保存. 由于以后用 QT4 改写代码后这些热键可能会大面积调整, 这里就不再一一描述. 如果想了解热键的实现方式, 请参考源文件 src/opengl_gui/callback.cpp 中的两个函数: keyboard_surface()和 keyboard_contour().