**课程设计指导书**

导读：本教程仅供初学者参考。通过一个简单的例子来介绍Quartus和modelsim的使用。例子如下：开发板上有10个LED灯，点亮4个并依次向左移动，形成流水灯。

目录

[**一、** **新建工程** 3](#_Toc317103283)

[1、点击File中的New Project Wizard选项，如下图。 3](#_Toc317103284)

[2、新建工程对话框 3](#_Toc317103285)

[3、添加文件对话框 3](#_Toc317103286)

[4、选择FPGA开发板型号 5](#_Toc317103287)

[5、选择EDA工具 6](#_Toc317103288)

[6、新建工程结束 7](#_Toc317103289)

[**二、** **设计输入** 8](#_Toc317103290)

[A、 选择其它编辑器，如UltraEdit 8](#_Toc317103291)

[B、 直接在Quartus里编写代码 8](#_Toc317103292)

[**三、** **编译** 10](#_Toc317103293)

[**四、** **仿真(modelsim)** 11](#_Toc317103294)

[1、写测试文件testbench.v 11](#_Toc317103295)

[2、新建一个sim.do文件，输入以下命令并保存 12](#_Toc317103296)

[3、开始仿真 13](#_Toc317103297)

[**五、** **下载到开发板上运行** 15](#_Toc317103298)

[1、 引脚分配 15](#_Toc317103299)

[**方法一**：用鼠标一个一个分配（较烦琐） 15](#_Toc317103300)

[**方法二**：使用tcl脚本文件（更便捷） 16](#_Toc317103301)

[2、 编译、下载 18](#_Toc317103302)

[**六、** **SignalTap的使用** 20](#_Toc317103303)

[1、 点击Tools中的打开SignalTap窗口如下图所示： 20](#_Toc317103304)

[2、 设置参数 21](#_Toc317103305)

[A、 添加欲观察信号 21](#_Toc317103306)

[B、 设置触发条件 21](#_Toc317103307)

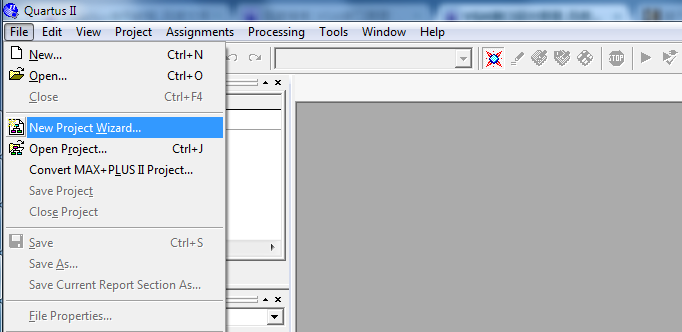
[C、 设置Signal Configuration面板 22](#_Toc317103308)

[3、 编译、下载 23](#_Toc317103309)

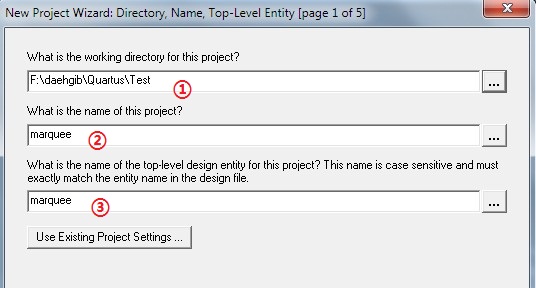
[4、 Run analysis 23](#_Toc317103310)

1. **新建工程**

1、点击File中的New Project Wizard选项，如下图。



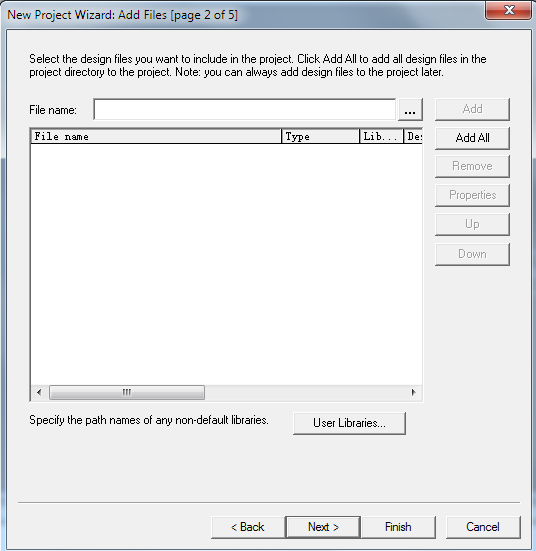
2、新建工程对话框



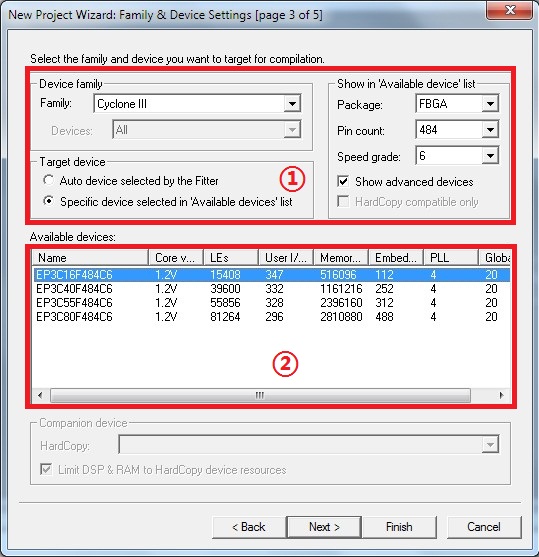
上图中，①指定所新建工程的存放路径，②是工程名，③是该工程中的顶层模块名（顶层模块就是端口与FPGA中的引脚相关联的那个模块，相当于C语言中的main入口函数），这个名字比较重要，若填错会直接导致后面的编译错误。默认情况下③和②名字是相同的。填写完后点击Next，进入添加文件对话框。

3、添加文件对话框

若已事先写好了RTL级代码，可以通过下面这个对话框添加相应文件。若没有则跳过。



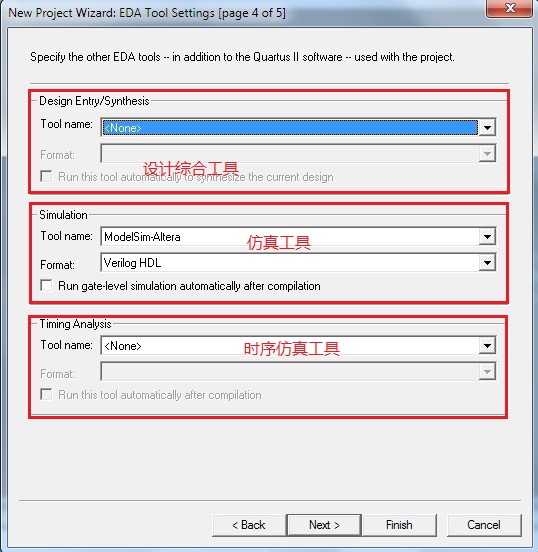
4、选择FPGA开发板型号



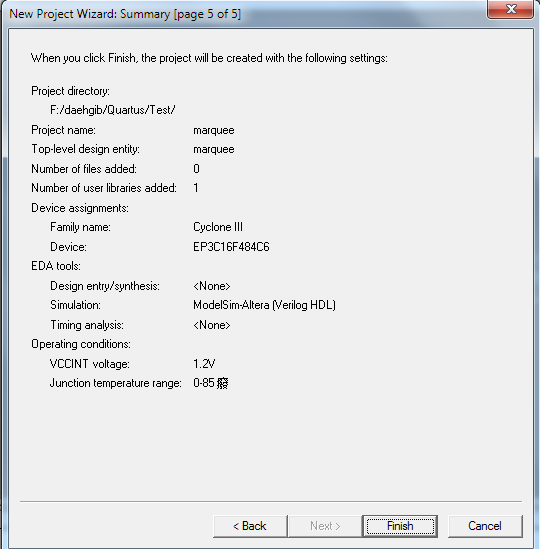
如果你现在还没有FPGA开发板，只是想先了解、熟悉Quartus开发环境，那你可以直接点击Next跳过这一步骤。如果你已经拥有一个FPGA开发板，想亲手点亮上面的LED灯，那你应该好好填写这一对话框。方框①中的各参数是筛选条件，方框②列出筛选结果。①中各参数可以参看你的开发板手册或产品说明书，依次选择完器件系列、封装类型、引脚数目和速度等级即可选择出你的开发板。本人开发板为EP3C16F484C6，选中后点击Next。

5、选择EDA工具

下面这个对话框是选择本次设计中要用的各种EDA工具，如果初次接触或者没有什么特别偏好默认即可，将会使用Quartus自带工具。点击Next就到最后一个对话框了。



6、新建工程结束



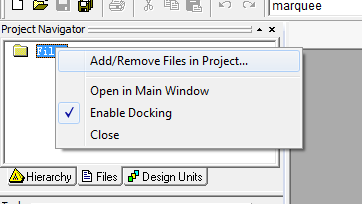
上图列出了该工程的各种设置选项，如果检查无误则点击Finish即完成了一个工程的创建。

1. **设计输入**

——根据个人偏好选择适合自己的代码编辑器。

1. 选择其它编辑器，如UltraEdit

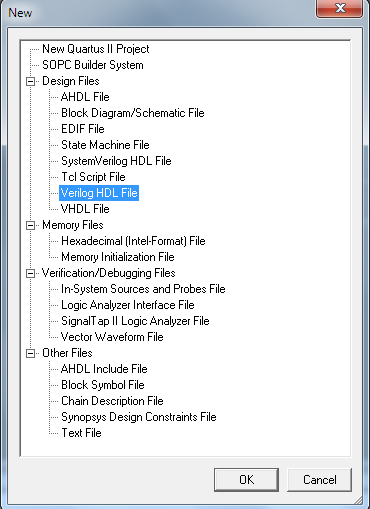
编写好代码后保存为.V文件并将其添加进工程即可。



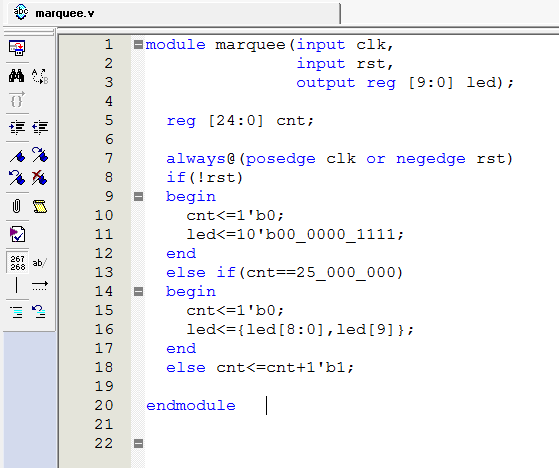
如上图，在工程浏览器的Files图标上右击选择Add/Remove Files in Project选项，弹出添加/移除文件对话框，将编辑好的RTL代码文件添加进去即可。

1. 直接在Quartus里编写代码

选择File中的New选项，弹出新建对话框如下：



新建对话框里有好多种文件类型，因为我们是用Verilog语言写代码，因此选择Verilog HDL File，点击OK即新建了一个.V文件。接着输入设计代码，注意别忘了保存文件。本例代码输入如下：



1. **编译**

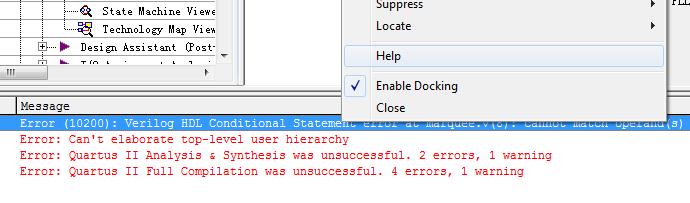
点击下图红框中的三角形对该工程进行编译。



若编译出错，根据错误提示修正后重新编译。编译成功后尽量减少Warning数目，尤其是Critical Warning。

**PS：关于Quartus的Error及Warning**

对于比较简单的语法错误，只要认真地查看错误提示是很容易发现问题并改正的。若某些问题实在理解不了，不知错误在哪，那可以进一步查看错误说明。如下图，选中理解不了的错误，右击再选择Help项即可打开本地网页文件，此网页是该错误更详细的说明，这样绝大部分问题都能理解并解决。希望大家遇到错误和警告不要急着找度娘、谷歌，更不要敷衍了之，而应多参看相关的Help页面，将其读懂才能有所提高。



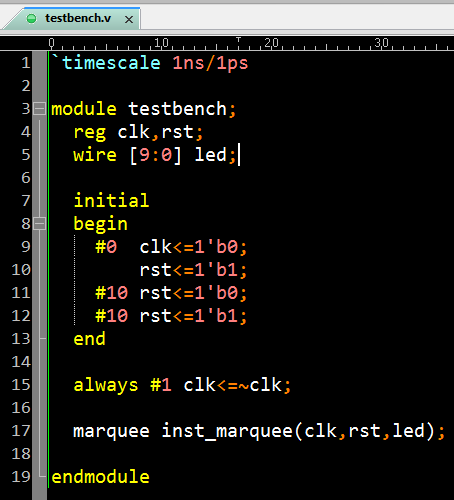
1. **仿真(modelsim)**

将目标文件下载到开发板之前进行仿真是很有必要的，仿真可以验证设计的功能是否达标，若出现了问题也能通过查看波形更好的解决。使用modelsim仿真的传统流程网上有很多相关教程，个人感觉操作比较烦琐。下面介绍使用modelsim中的**脚本文件**进行仿真的方法。

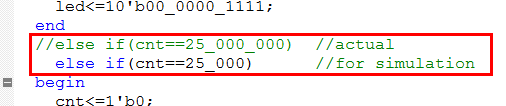
该方法不用新建工程，只需在modelsim的Transcript窗口敲击命令即可，省去了大量烦琐的鼠标操作，尤其是在调试一个工程时，可节省大量时间。

1. 写测试文件testbench.v

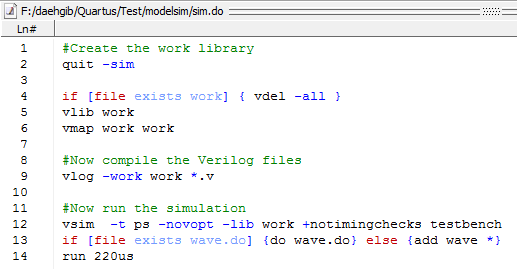
写测试模块跟写marquee.v是差不多的，这里就不介绍测试模块的具体细节了。本例测试模块代码如下图所示：



**注意**：有些设计的信号变化间隔是比较大的，如本例的led信号是每隔0.5s变化一次，0.5s相对于仿真时间间隔来说实在太大了，因此仿真之前先更改分频always块的分频系数，最好是写两条语句（分别使用不同的分频系数），一条用于仿真、一条用于实际运行，使用’//’屏蔽其中一条，如下图所示：



2、新建一个sim.do文件，输入以下命令并保存



大家可能还看不懂上面各条语句都是什么意思，下面简单介绍下：

**、**

**2 quit –sim：退出当前仿真状态**

**4 if [file exits work] { vdel–all }：如果work文件夹存在，则删除work单元库（即删除work文件夹）。**

**5 vlib work：新建一个名为work的库**

**6 vmap work work：将逻辑意义上的work库映射到实际文件夹work上**

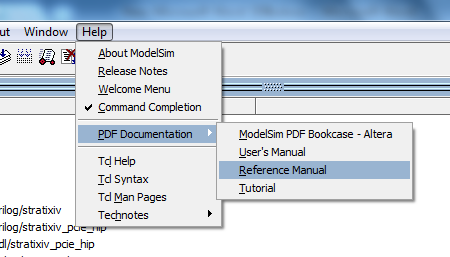
**9 vlog–work work \*.v：将本级目录下的所有.v文件编译到work库中**

**12 vsim–t ps–novopt–lib work +notimingchecks testbench：进行仿真。-t ps指定仿真的时间精确度；-novopt禁止modelsim对各模块进行优化；-lib work指定当前工作库，即从work库里查找仿真要调用的模块；testbench就是测试模块名，为了使该.do文件更具通用性，最好以后每次写测试模块都起名为testbench。**

**13 if [file exists wave.do] {do wave.do} else {add wave \*}：该语句的作用是添加波形，如果wave.do(稍后介绍)文件存在，则执行wave.do文件；否则添加testbench模块中的所有端口。**

**14 run 220us ：仿真执行220us**

以上只是这些用到的脚本的简单说明，上面这些命令还有好多其它参数及用法，如果对这些命令比较感兴趣，可以参看modelsim中help里面的pdf文件，如下图所示：



3、开始仿真

所有文件都准备好了，下面可以开始仿真了。注意各文件的保存位置，marquee.v、testbench.v、sim.do文件均保存在同一目录下。

1. 启动modelsim，首先得将modelsim的工作路径转移到上述三个文件所在目录中，假设上述三个文件在F:\daehgib\Quartus\Test中，那在modelsim的Transcript窗口输入以下命令即可转移当前路径。

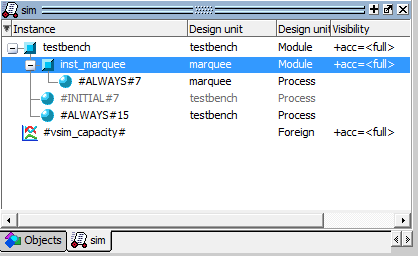
**cd f:/daehgib/quartus/test（注意是’/’而不是’\’）**

注：输入dir可以查看当前目录下的文件，输入路径时可按tap键进行匹配。使用方向键中的“上”、“下”可以调用之前执行过的命令。

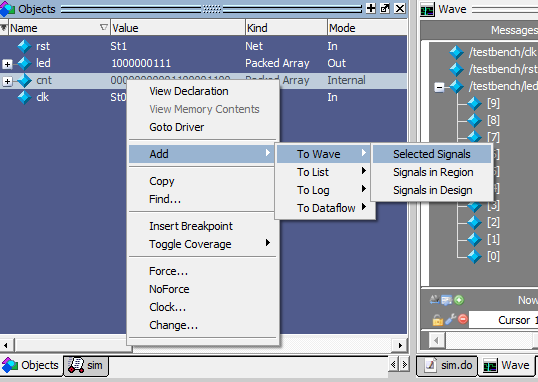
1. 输入do sim.do命令开始仿真

不出意外的话，马上就可以看到仿真波形了。默认情况下所显示的波形是测试模块testbech中声明的所有信号，如果你还需要观察其它信号波形，那么可以手动添加。如本例中我要观察marquee中的cnt信号，添加方法如下：

1. 先在sim窗口中选中欲添加信号所处的模块，我要添加cnt信号，而cnt信号在marquee模块中，因此我选中marquee模块，如下图：

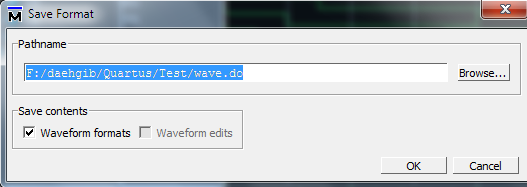


1. 然后切换到旁边的Objects窗口，此时你可以看到cnt信号了，鼠标放上去并右击，依次选择Add->To Wave->Selected Signals即完成信号的添加，如下图：



1. 你现在是不是想立即执行do sim.do命令进行再次仿真以观察cnt信号？慢着！！！如果这样你就功亏一篑了，你可以试一下，你将会发现重新执行后显示的波形和第一次是一样的，因为你没有把修改后的波形保存下来。因此，剩下一步操作就是保存波形文件。方法如下：

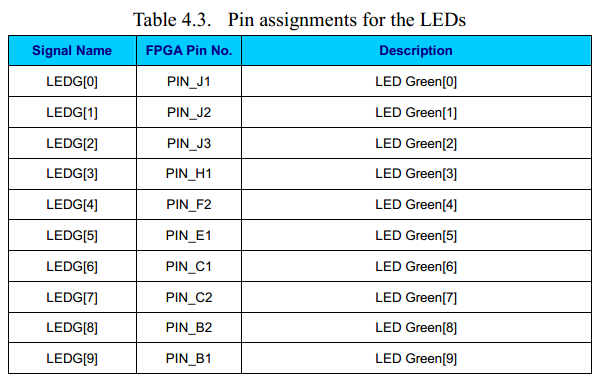
**点击File中的Save Format**后弹出如下窗口：



Wave.do就是之前所说的那个文件，现在理解sim.do文件中第13行的那条命令了吧！使用默认路径，wave.do将会保存在和sim.do文件同一目录下。每次修改波形后都执行一下保存操作，新的wave.do会覆盖掉之前的版本，以便下次仿真后显示出希望观察的波形。

1. **下载到开发板上运行**
2. 引脚分配

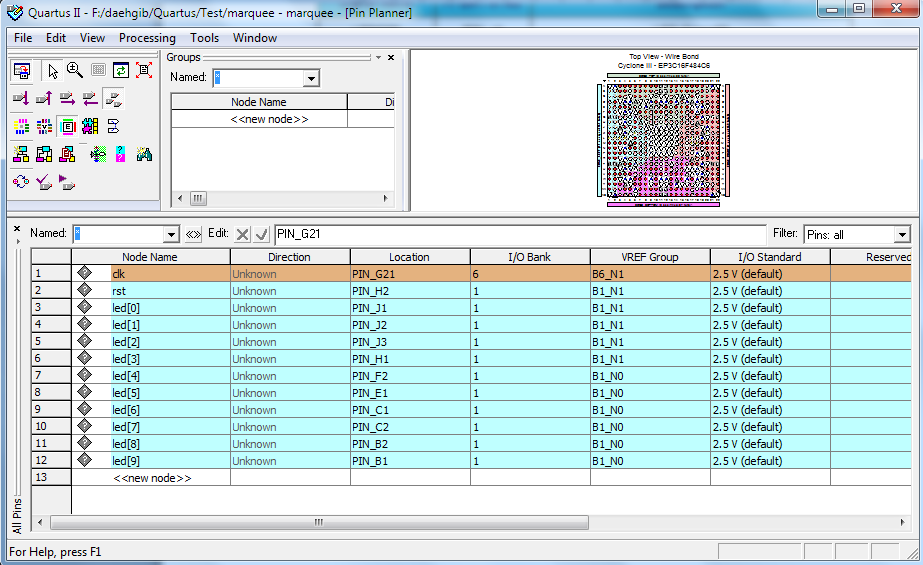
有两种方法，但前提都是知道FPGA开发板各引脚编号，可参看开发板手册。下图截自于开发板说明文档。



引脚编号就是指上图中的第二列字母，如LED[0]即第1个LED对应引脚编号为PIN\_J1，其它类似。更多内容其参看各自开发板的说明文档。

**方法一**：用鼠标一个一个分配（较烦琐）

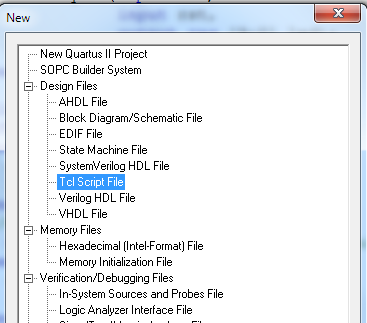
1. 点击Assignments中的Pins弹出引脚分配窗口，如下图：



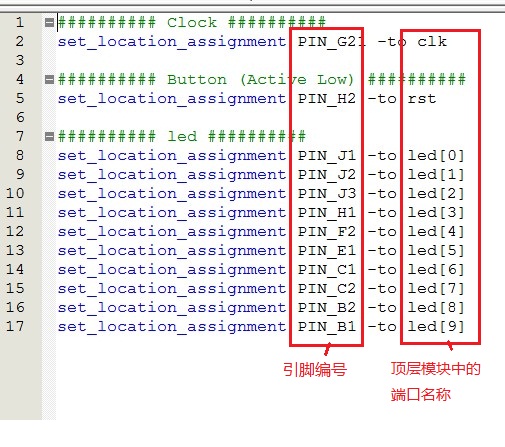
1. 依次在Location列填入各信号相应的引脚编号

**方法二**：使用tcl脚本文件（更便捷）

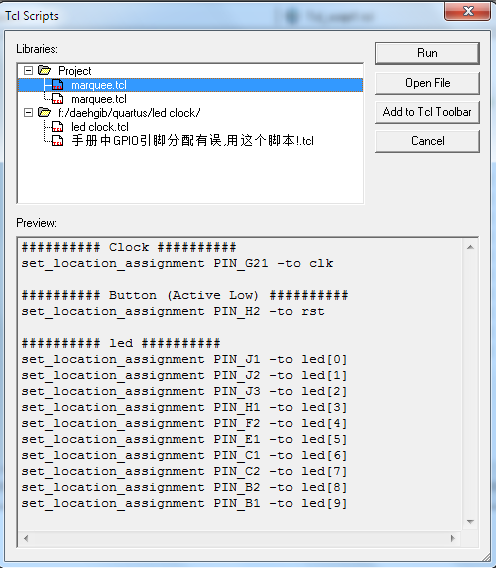
1. 点击File中的New，再选择Tcl Script File后点OK，新建一个tcl文件。



1. 按如下格式输入各引脚对应编号（最好是将开发板的所有引脚编号按下面这种格式写好后保存在一个文本文件中，以后要输入时只需部分COPY）。



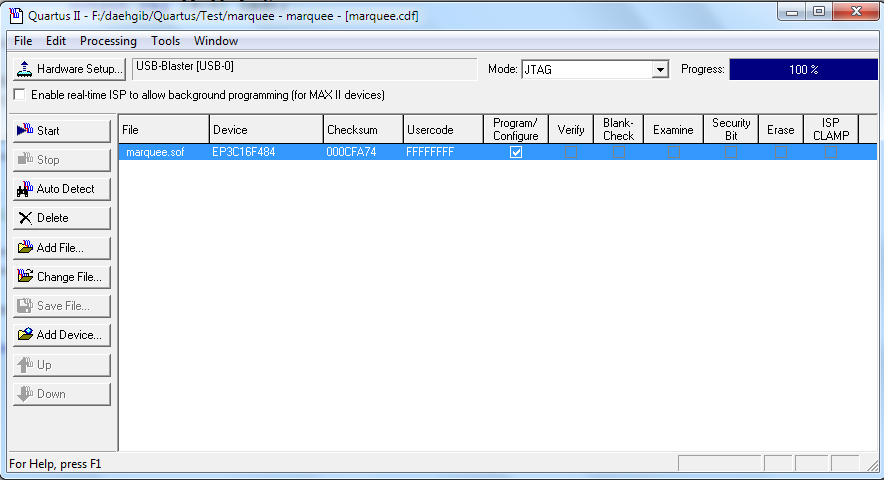
1. 全部输入完成后，保存为marquee.tcl。点击Tools中的Tcl Scripts后弹出运行脚本窗口，如下图所示：



选择刚刚保存的marquee.tcl后点击Run，成功运行后即完成引脚分配。可以点击Assignments中的Pins弹出引脚分配窗口查看分配情况，以确认引脚分配无误。

1. 编译、下载

点击三角形图标再次编译，编译成功后先连接上FPGA开发板，驱动的安装自己解决（驱动程序在X:\altera\91\quartus\drivers\usb-blaster里）。然后点击红框中的Programmer图标，弹出下载窗口如下图所示：



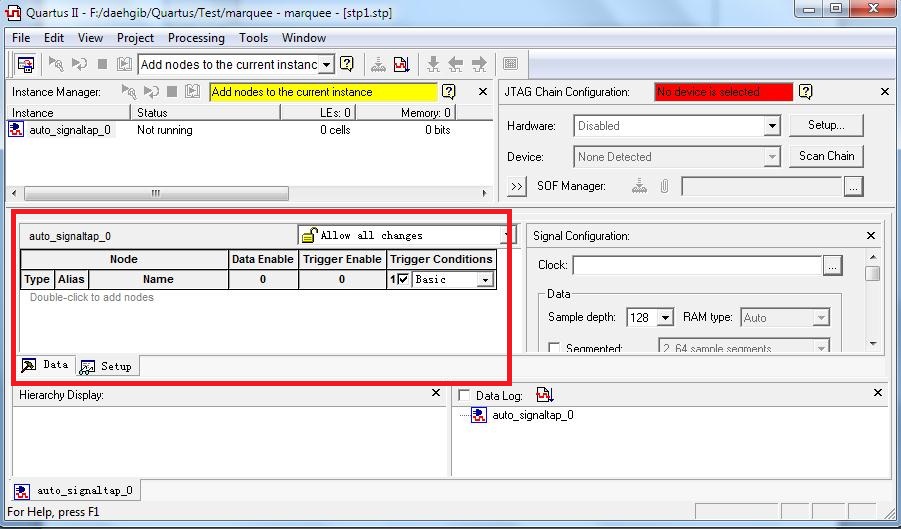
在连接无误的情况下选择marquee.sof点击Start即可将电路下载进FPGA中，如操作正确则看到开发板上的LED如流水般闪动。

1. **SignalTap的使用**

SignalTap是Quartus的在线调试工具，通过它可以实时观察到开发板上各引脚信号的变化，方便开发者发现问题并解决问题。下面介绍该工具的具体使用方法：

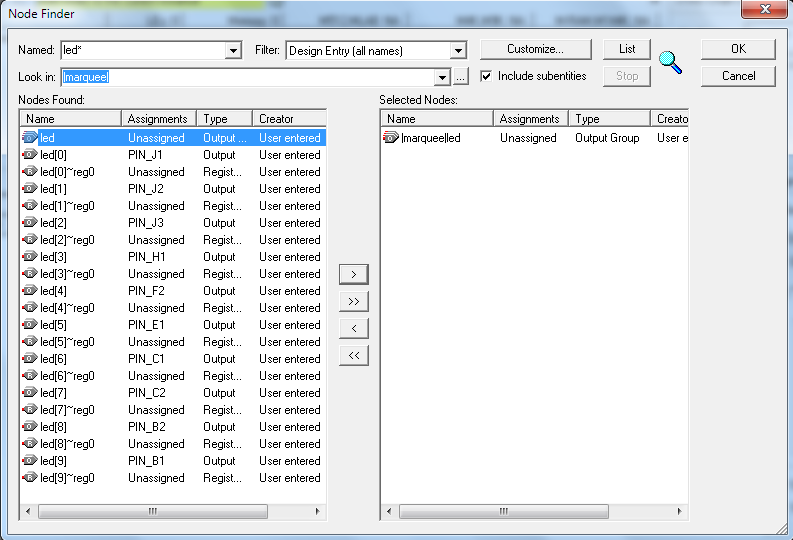
首先应当**强调**的还是**分频系数**的问题，由于采样缓存也就那么大，储存的数据是有限的，为了去掉不必要的数据，缩小时间间隔有利于充分利用缓存，从而更直观地展示出波形的变化。

1. 点击Tools中的打开SignalTap窗口如下图所示：



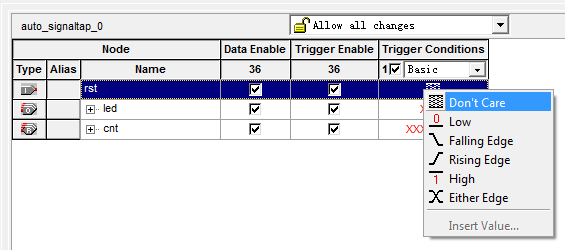
1. 设置参数
2. 添加欲观察信号

在上图红框处双击弹出节点查找对话框，如下图所示：



上图所示对话框会在Quatus好多地方出现，都是为了查找端口、节点或引脚，用法都一样。相信大家摸索一下就知道如何使用了。这里我们找到要观察的信号rst、cnt、led并添加进去。

1. 设置触发条件



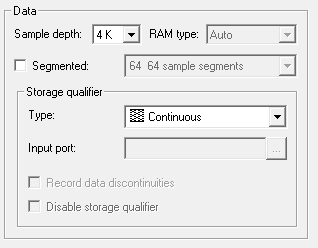
如上图所示，第四列为触发条件，有Basic和Advanced两种类型。在Basic模式下，鼠标放上去并右击可选择无关、低电平触发、下降沿触发、上升沿触发、高电平触发、边沿触发，对于位宽大于1的信号还可以设置具体值触发。在Advanced模式下，可以设置更复杂的条件进行触发，这里就不再讨论了。本例中三个信号均选择无关，即触发条件永远满足，即时采集数据并显示波形。

1. 设置Signal Configuration面板
2. **Clock**



设置采样时钟，这里同样要用到前面介绍的节点查找对话框，使用方法一样。采样时钟就是确定SignalTap以多大的频率对引脚信号进行采样，然后进行触发条件的匹配。注意采样时钟不能为2中添加的欲观察信号，并且采样时钟的频率要大于欲观察信号的最高频率。本例设置为开发板上产生的系统时钟。

1. **Data**

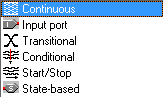


**Sample depth：**设置采样缓存的大小，越大能储存的数据就越多。但越大所需综合时间也越久，并且可能会超过FPGA的额外资源（即综合不了）。

**RAM type：**设置采样缓存的存储器类型，这个只有当FPGA器件含有多种存储器类型时才有效。

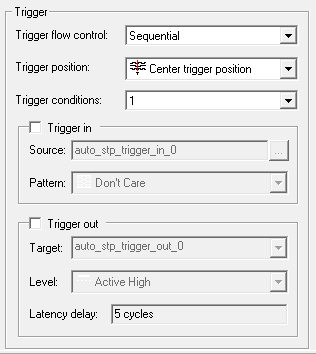
**Segmented：**选择是否将采样缓存进行分段。如表示将4K的采样缓存分为4段，每段为1K。若启用该模式，则每当触发条件满足一次就捕获一段数据，这样可以去掉一些无关数据。

**Storage qualifier：**设置数据存储条件。默认情况下是在采样时钟的边沿连续采集数据并储存。Type有如下几种类型：



各种情况自己去摸索一下就知道有什么区别了，这里就不一一介绍了。本例设置最简单的情况，即Continuous，连续存储数据。

1. **Trigger**



**Trigger flow control：**选择Sequential，另一个本人也没用过，不知道啥意思。

**Trigger position：**触发状态在存储的整段数据中所处的位置。有以下三种类型：

Pre trigger position： 88％触发前数据，12％触发后数据；

Center trigger position： 触发前后数据各占50％；

Post trigger position： 12％触发前数据，88％触发后数据；

**Trigger conditions：**触发条件的级数，默认为一级，最高10级。

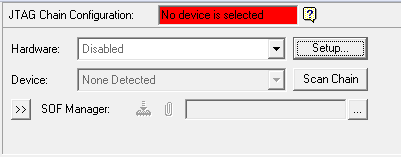
**Trigger in & Trigger out：**很惭愧，这个本人还是没用过，使用默认值吧。

1. 编译、下载

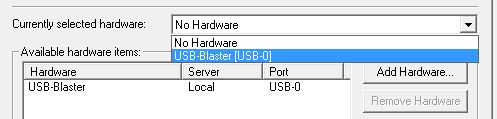
完成2中各参数的设置后，选择File中的save保存为marquee.stp。然后回到quartus主界面，点击三角形重新编译，编译成功后将marquee.sof下载到开发板中。

1. Run analysis

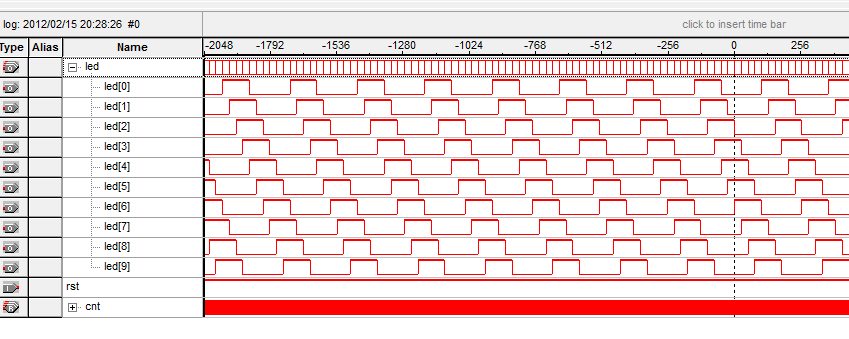
再回到SignalTap界面，先观察右上角面板的状态。若显示如下，则表示还未检测到开发板。



点击上图中的Setup弹出对话框如下：



选中USB-Blaster后点Close，SignalTap会再次检测器件，连接成功后点击黑框中的三角形，SignalTap开始采样数据，然后就会出现如下波形：



好了，SiganalTap的基本用法就这些了。还有其它参数、其它选项没用到过的就自己摸索了吧！另外，学会使用软件中的Help，Help是最原始的资料。好多问题都能在里面搜索到。如下：

