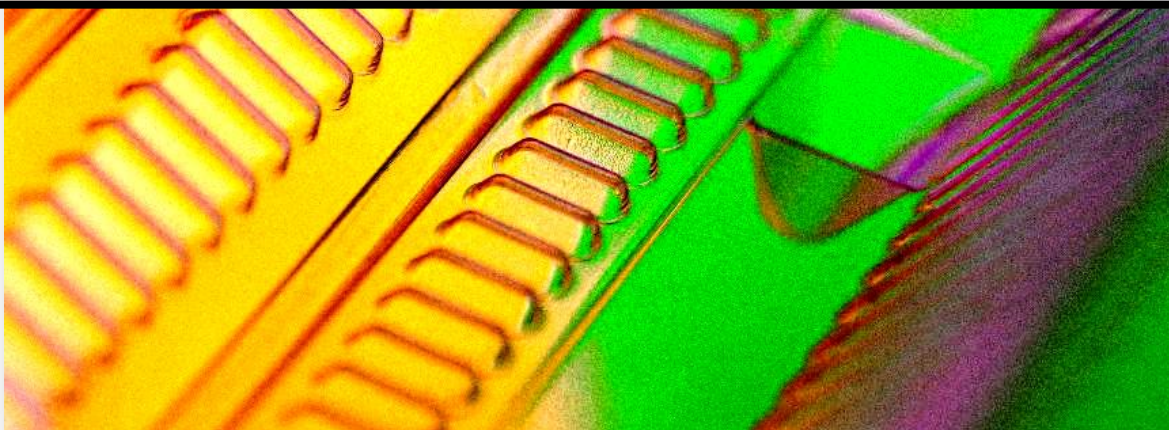


zrtech

**FPGA/CPLD 开发套件实验教程**

**--仿真，调试，设计篇**



**WWW.ZR-TECH.COM**

## 实验五、SignalTap II 嵌入式逻辑分析仪

### 实验目的：

通过这个基础实验，使用户了解 SignalTapII 的基本原理，熟悉 SignalTapII 结合 QuartusII 软件的基本调试方法。

### 实验结果：

使用 SignalTapII 调试测试工程。

### 实验原理：

#### 1. Quartus II 在线调试工具简介

Altera 在 Quartus II 工具提供了五种不同的在线调试方法。这里的在线调试是指协助或不借助于外部工具的 FPGA 板级调试。这些方法调试形式上稍有不同，互有优劣，目的都是为了帮助设计者更有效的完成板级验证。至于在面对这些方法时如何选择更适合特定的工程，应该综合考虑设计者的经验、喜好、对工具的熟悉程度、器件支持与否与工程的调试需求等因素

##### **SignalProbe**

信号探针方式不影响原有的设计功能和布局布线，只是通过增加额外布线将需要观察调试的信号连接到预先保留或者暂时不使用的 I/O 接口。该方式相应得到的信号电平会随布线有一定的延时，不适合于高速、大容量信号观察调试，也不适合做板级时序分析。它的优势在于不影响原有设计，额外资源消耗几乎为零，调试中也不需要保持连接 JTAG 等其他线缆，能够最小化编译或是重编译的时间。

##### **SignalTap II Embedded Logic Analyzer**

SignalTap II 在线逻辑分析仪很大程度上可以替代昂贵的逻辑分析仪，为开发节约成本；同时也为调试者省去了原本繁琐的连线工作，而有些板级连接的外部设备很能观察到的信号都能够被轻松的捕获。如果对设计进行模块的区域约束，也能够最小化使用在线逻辑分析仪对设计带来的影响。在线逻辑分析仪的采样存储深度和宽度都在一定程度上受制于 FPGA 器件资源的大小。使用该方式必须通过 JTAG 接口，它的采样频率可以达到 200MHz（若器件支持）以上，而不用像外部调试设备一样担心信号完整性问题。

##### **Logic Analyzer Interface**

这里的逻辑分析仪接口针对于外部逻辑分析仪的。调试者可以设置 FPGA 器件内部多个信号映射到一个预先保留或者暂时不使用的 I/O 接口上，从而通过较少的 I/O 接口就能够观察 FPGA 内部的多组信号。

##### **In-System Memory Content Editor**

在线存储内容编辑是针对设计中例化的内嵌存储器内容或常量的调试。可以通过这种方式在线重写或者读出工程中的内嵌存储器内容或常量。对于某些应用可以通过在线更改存储器内容后观察响应来验证设计，也可以在不同激励下在线读取当前存储内容来验证设计。总之，这种方式对存储器的验证是很有帮助的。

### In-System Sources and Probes

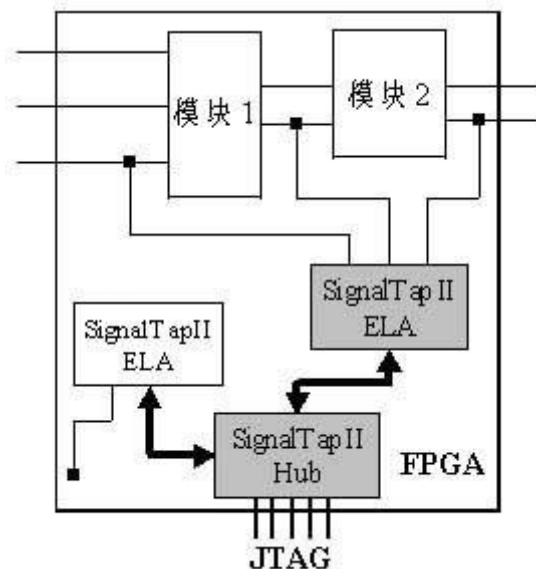
这种方式是通过例化一个定制的寄存器链到 FPGA 器件内部。这些寄存器链通过 JTAG 接口与 Quartus II 通信，它又能够驱动 FPGA 器件内部的某些输入节点信号，采样某些输出节点信号。这就使得调试者不用借助外部设备就能够给 FPGA 添加激励并观察响应。

这五种调试工具，我们目前着重给大家介绍 SignalTap II 与 In-System Memory Content Editor，这两种是平时设计中最常用也是最好用的调试工具，本节主要介绍 SignalTap II，In-System Memory Content Editor 将在下节介绍。

## 2. SignalTap II 介绍

随着设计复杂度的增加，传统的设计验证方法需要其他技术和工具的补充，因为这些可编程芯片系统（SOPC）进行完全的仿真模拟是不可在适当的时间内完成。而且，设计规模的陡增必然需要新的工具来观察已编程期间的内部操作。尤其是随着第三方 IP 使用的增加，它们需要获取内部探测来验证操作，使其和设计的其他部分相分离。最后，由于封装技术的提高，必须开发新的方法以便对日趋小型化和大规模封装的硬件进行验证。

幸好 Quartus II 支持 SignalTap II 以满足这些需求，如下图所示。SignalTap 允许设计者在 FPGA 运行期间同时监视内部信号。通过下载电缆或传统的分析设备连接到用户的 PC 板卡上，便可以观察到这些信号的波形。使用 SingnalTap 就类似于使用逻辑分析仪，能够设置初始化、触发（内部或外部）和显示条件以及观察的内部信号，用户以此可以研究设计的运行状态。用户的分析参数可以被编译为嵌入逻辑分析仪（ELA），它和设计的其他数据一起配置 FPGA。Altera 全系列 FPGA 器件支持 SignalTap，采用 Byteblaster II 或者 USB blaster 作为器件的下载电缆。



若没有采用 SignalTap 接口，用户必须更改设计以探测内部逻辑的连线。设计的内部连线必须连接到顶层设计的管脚上。如果结点处于庞大分级设计的下层，那么改起来很复杂，同时很耗时，而且破坏了设计的完整性。ELA 接口支持拖放选择用于逻辑分析的连线。这个接口根本就无需改变设计。选择了 ELA 的输入通道之后，需要重新编译设计把 ELA 配置加入期间

配置文件中。重新编译只是把一个 ELA 实例添加到现有的设计中, 而无需改变已有的设计。更新后的配置文件重新配置器件后, 标准逻辑分析仪就会可以检测那些被连接到器件管脚的内部信号了。

输入通道的样值存储在器件的嵌入存储块内, ELA 功能监测输入通道是否发生触发事件。一旦 ELA 存储了满足触发状态的足够数据, ELA 停止采样监测输入通道。然后数据上载到主机, 显示在 Quartus 的波形编程器中。数据的主载速率取决于 JTAG TCK 信号的速率。ELA 功能会使用设计本身占用以外的器件资源。ELA 是可参数化的, 因此能够使用有效的资源。

SignalTap II 支持以用户指定的格式识别和显示总线使所捕获的数据更加易懂。SignalTap II 嵌入式逻辑分析仪能够以等价的十六进制、无符号十进制、二元补码形式的符号十进制, 符号大小表示法表示的符号十进制、八进制、二进制、8 比特 ASCII 等格式来显示总线。用户还可以选择条形图或者线性图表示总线时间关系。

SignalTap II 支持多文件输出数据结果, 嵌入式逻辑分析仪可以采用矢量波形 (vwf)、矢量表 (tbl)、矢量文件 (vec)、逗号分割数据 (csv) 和 Verilog 数值更改转存 (vcd) 文件格式输出所捕获的数据。这些文件格式可以被第三方验证工具读入, 显示和分析 SignalTap II 嵌入式逻辑分析仪所捕获的数据。由于 signaltap 需要占用大量的存储器与逻辑资源, 所以一般的小容量 CPLD 是不支持 SignalTap II, 本实验也仅限 FPGA 的核心板, 如 CORE2-5U, CORE2-5SD 等。

### 3. 使用 SignalTap II 操作流程

若要使用 SignalTap II 逻辑分析器, 必须先建立 SignalTap II 文件 (stp) 此文件包括所有配置设置并以波形显示捕获到的信号。一旦设置了 SignalTap II 文件, 就可以编译工程, 对器件进行编程并使用逻辑分析器采集和分析数据。

以下步骤描述设置 SignalTap II 文件和采集信号数据的基本流程。

(1) 建立新的 SignalTap II 文件。

(2) 向 SignalTap II 文件添加实例, 并向每个实例添加节点。可以使用 Node Finder 中的 SignalTap II 滤波器查找所有预综合和布局布线后的 SignalTap II 节点。

(3) 分配一个采样时钟。

(4) 设置其他选项, 例如采样深度和触发级别等。

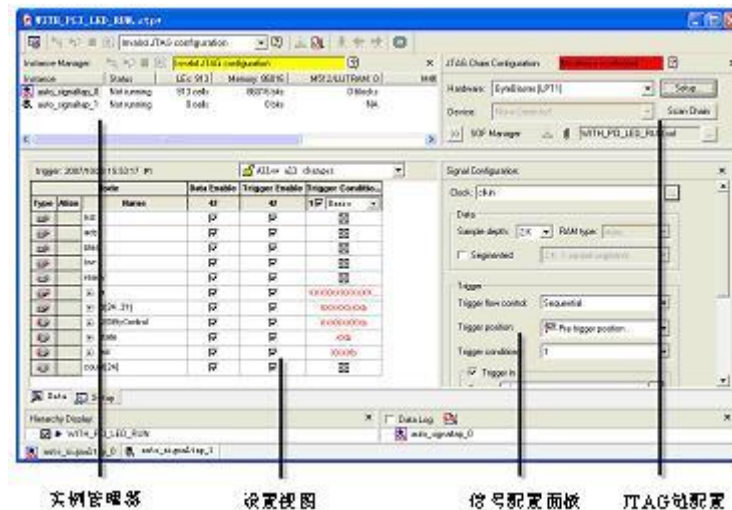
(5) 完全编译工程文件。

(6) 下载程序到 FPGA 中。

(7) 运行硬件并打开 SignalTap II 观察信号波形。

如图所示是 SignalTap II 逻辑分析仪的界面。





实例管理器

设置视图

信号配置面板

JTAG链配置

其中实例管理器对话框识别出设计中等待测试的所有验证过的逻辑分析仪，它们可以用来捕获并存储数据。该对话框还可以对用来生成每个分析仪的资源进行估算。信号配置面板用于设置采样信号和触发信号。采样时钟信号支持超过 200MHz 频率，采样深度最大高达 128K。数字示波器或逻辑分析仪中，触发器是一个重要的组成部分。触发器的性能很大程度上决定了仪器的性能。触发器是逻辑级别、逻辑边缘触发方式、逻辑样本等逻辑事件的组合。通过不同的触发方式实现对信号的不同采样,SignalTap 接口定义的触发事件中，每个事件的输入通道可以监测 10 个基本或高级的触发级别。触发级别向 SignalTap II 逻辑分析仪指明何时开始采集数据，10 个触发级别为设置复杂的触发条件提供了足够的灵活性，帮助验证工程师分离错误或者找出问题原因，如果设置了多触发级别，直到所有的触发条件顺序满足后，才开始采集数据。触发位置允许指定在选定实例中在触发器之前和触发器之后应采集的数据量。分段的模式允许通过将存储器分为周密的时间段，为定期事件捕获数据，而无需分配很深的采样深度。其中对环形缓冲模式支持 4 个触发位置；这样，当触发条件满足后，用户可以更多地控制应该捕获并显示什么样的数据。

- “前” 触发位置向软件表明，在达到触发条件前，保存所发生采样的 12%，达到触发条件后，再保存采样的 88%。
- “中” 触发位置向软件表明，在达到触发条件前，保存所发生采样的 50%，达到触发条件后，再保存采样的 50%。
- “后” 触发位置向软件表明，在达到触发条件前，保存所发生采样的 88%，达到触发条件后，再保存采样的 12%。
- “连续” 触发位置向软件表明，以环形缓冲的方式进行连续采样保存，直到用户中断为止。

## 具体步骤：

### 1.利用 SignalTap 进行基本数据的观测

由于 signaltap 是对我们的设计工程进行调试的，所以首先我们新建立一个 quartus 工程，加入一段 verilog 或者 vhd1 程序：  
Verilog 版本：

```
module LEDglow(clk, LED);
input clk;
output LED;
reg [25:0] cnt;
always @(posedge clk) cnt<=cnt+1;
wire [3:0] PWM_input = cnt[25] ? cnt[24:21] : ~cnt[24:21];    //ramp the PWM input up and down

reg [4:0] PWM;
always @(posedge clk) PWM <= PWM[3:0]+PWM_input;
```

VHDL 版本：

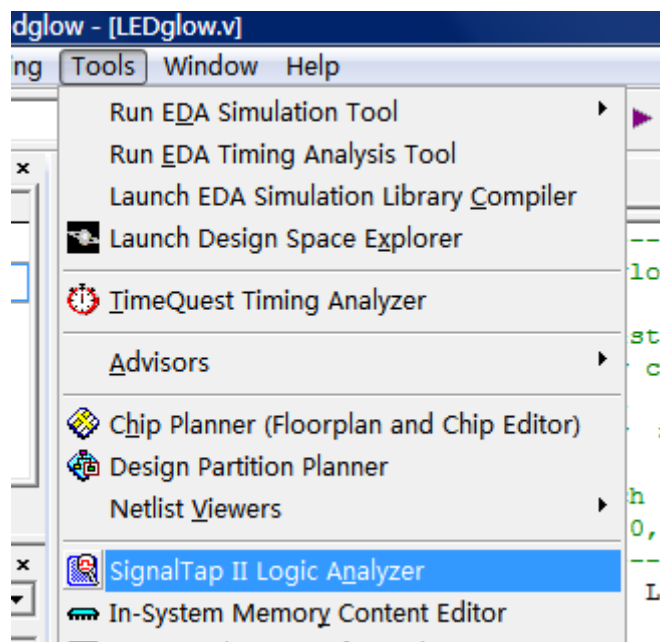
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity LEDglow is
port(clk:in std_logic;
      LED:out std_logic);
end LEDglow;

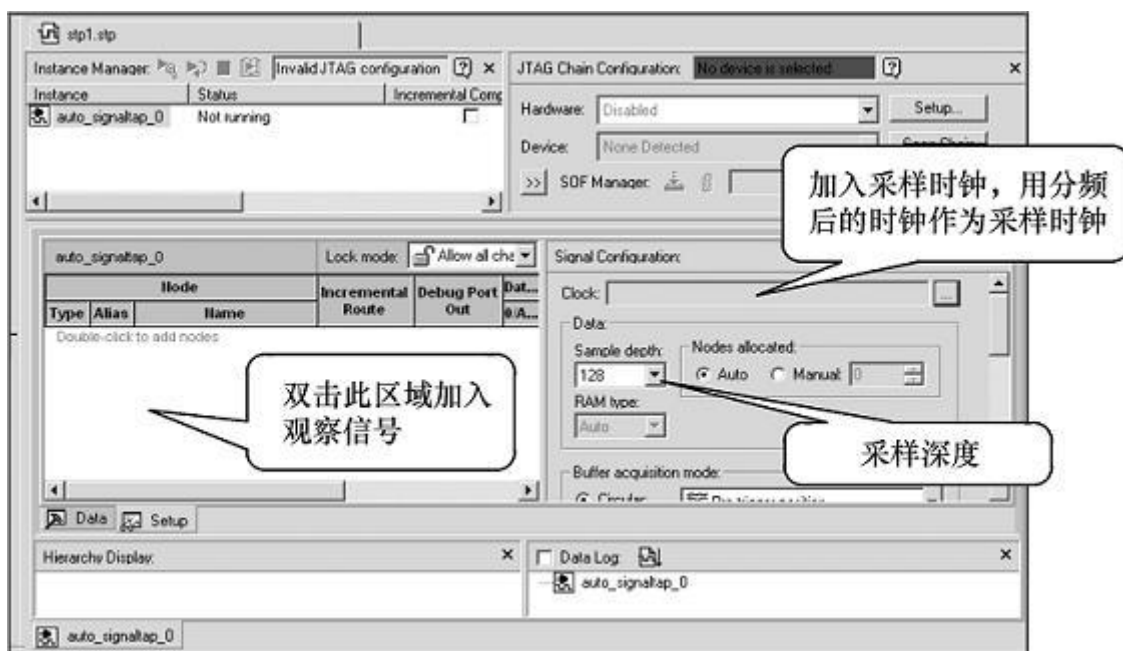
architecture fun of LEDglow is
signal cnt: std_logic_vector(25 downto 0);
signal PWM_input: std_logic_vector(3 downto 0);
signal PWM: std_logic_vector(4 downto 0);
begin
LED <= PWM(4);
PWM_input <= cnt(24 downto 21) when cnt(25)='1' else not cnt(24 downto 21);
--ramp the PWM input up and down
process(clk)
begin
if(clk'event and clk='1')then
cnt <= cnt + 1;
PWM <= '0' & PWM(3 downto 0) + PWM_input;
end if;
end process;
```

这个程序不难理解，即通过 PWM\_input 输入不同的值来控制计数器的一次增加的幅度，从而改变分频比，从而改变 LED 的亮度，每当计数器计满后，就改变一次分频比幅度的趋势，从而实现 LED 的渐亮—渐暗的呼吸灯效果。

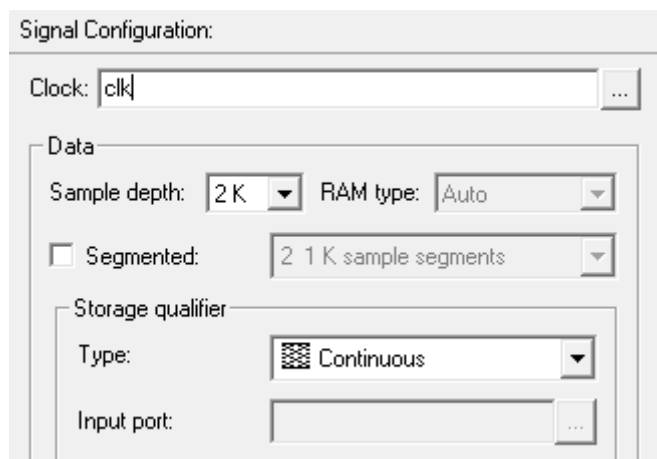
接下来重新编译一下工程，分配好管脚，也可以用 tcl 文件进行分配。再打开 signaltapII，在菜单中选择 “Tools” / “SignalTap II Logic Analyzer” 选项



打开如图所示主界面。其具体功能解释如下：



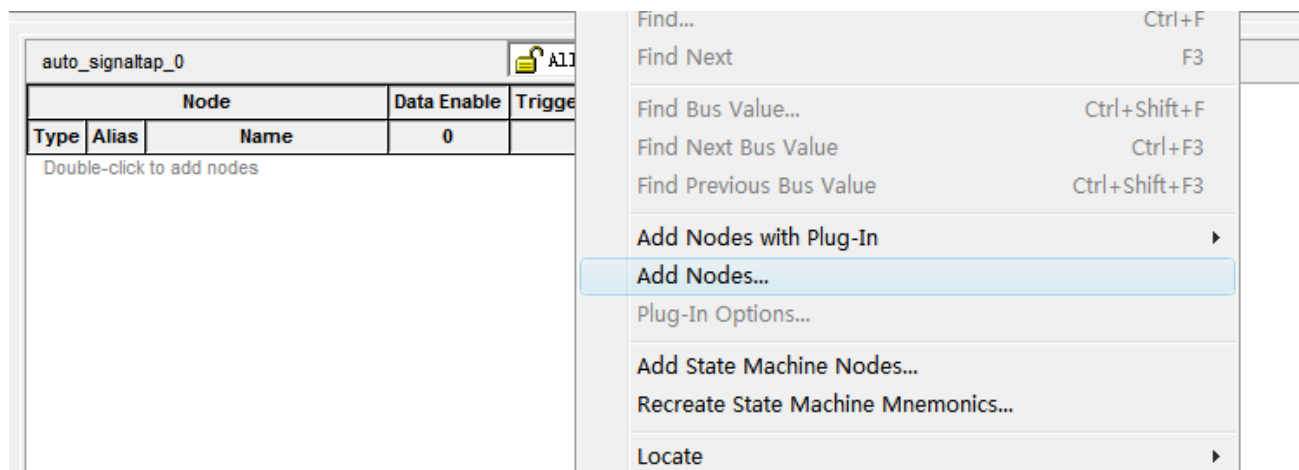
加入采样时钟，这里直接采用系统主时钟，采样深度设置为 2k。



### SignalTap II 基本设置需注意以下问题

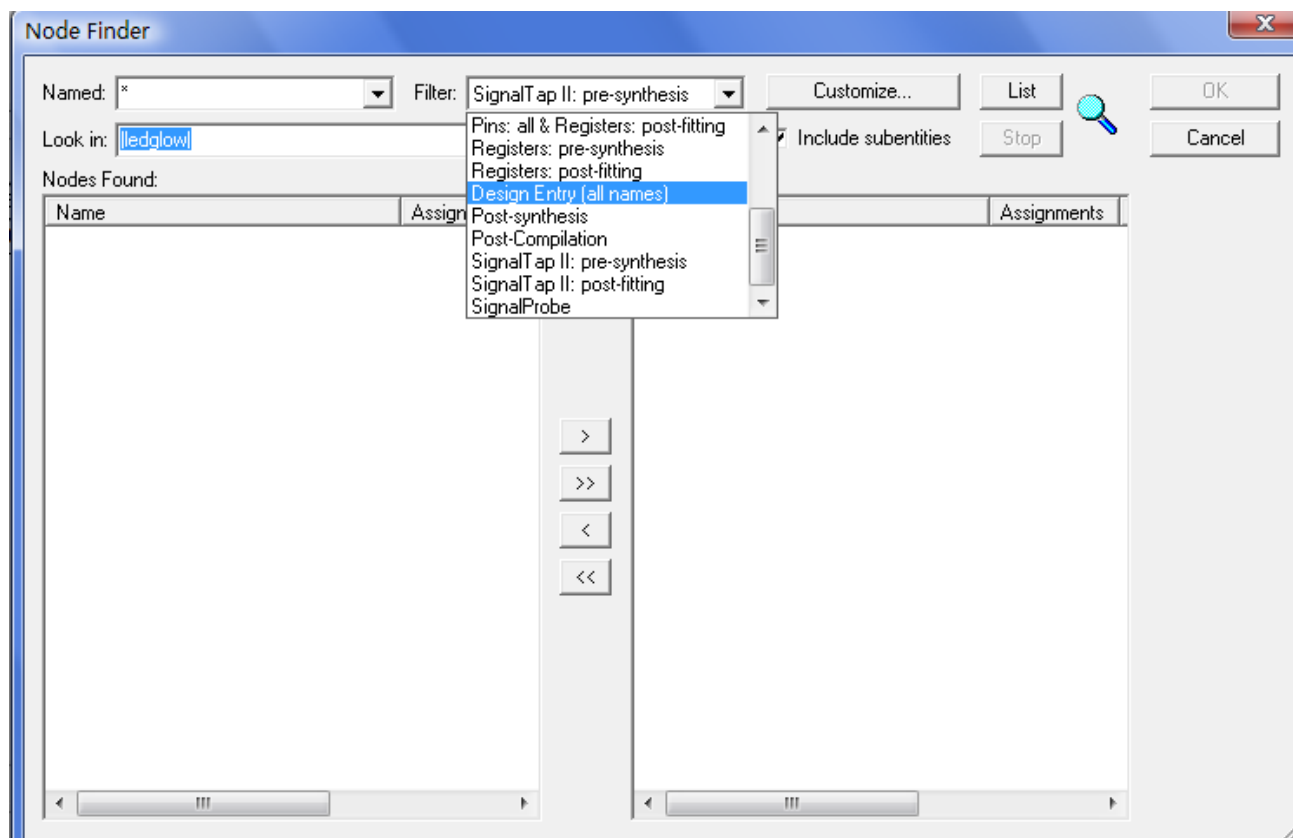
- (1) 采样时钟决定了逻辑分析仪的最小采样分辨率，如果你需要的精度越高，就需要高速时钟进行采样，反之，如果你需要观测较长的时间，那么最好用低速时钟采样。
- (2) 采样深度决定了采样的数据量，深度越深，消耗的 ram 越多，大家可以自己权衡。

在空白处双击或者点右键，加入采样需要观察的信号与寄存器

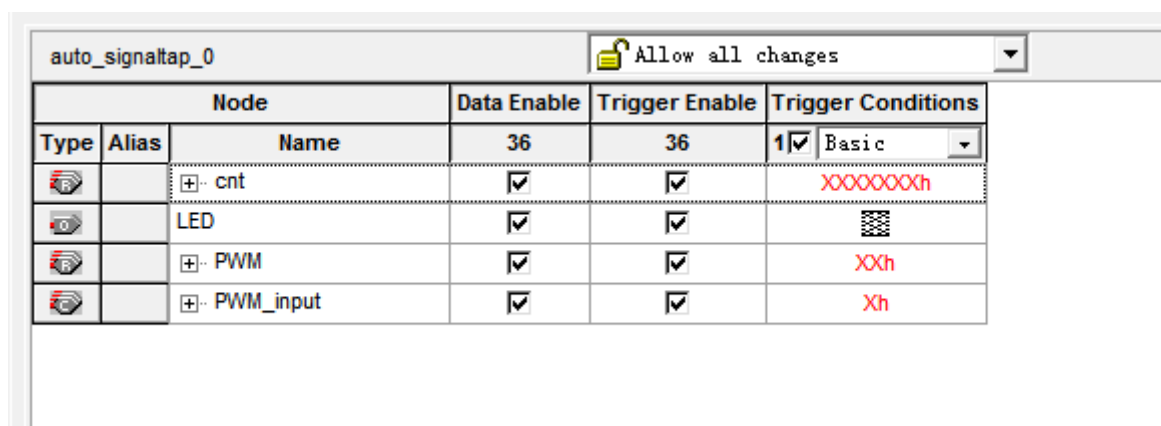


打开了 node finder，用法和 vwf 仿真时候一样，如果要显示所有的信号，寄存器，管脚，则可如下选择，然后点击 list，将感兴趣的信号添加到右侧

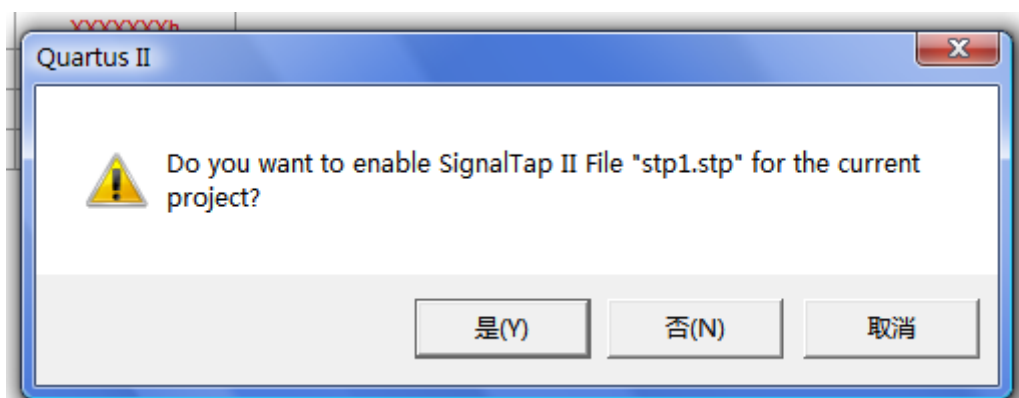




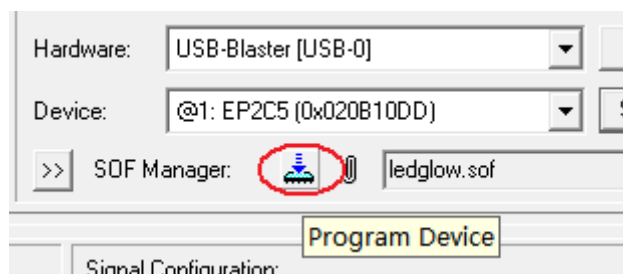
添加如下四个信号，注意由于 clk 已经作为采样时钟了，所以就不能再添加了



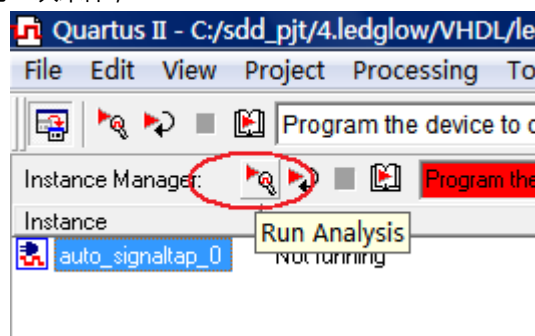
保存文件，出现如下窗口，选择“是”，启用刚才设置的 signaltap 文件



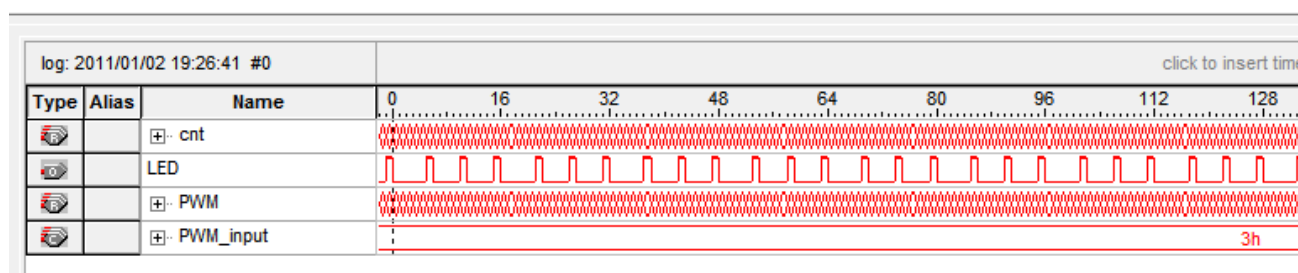
由于 signaltap 是嵌入在 FPGA 中的，所以需要重新编译，加入 signaltap 的相关逻辑与存储，所以我们整体编译工程。之后使用 JTAG 模式将程序下载到 FPGA 中运行。即再次打开 SignalTap II，在右上角选择合适的下载线缆与 sof 文件，然后点击 Program Device 下载 sof 文件。注意由于 SignalTap 页面下已经集成了 Quartus 的 programmer，所以在这里下载与在 quartus 中的 programmer 下载是完全一样的。



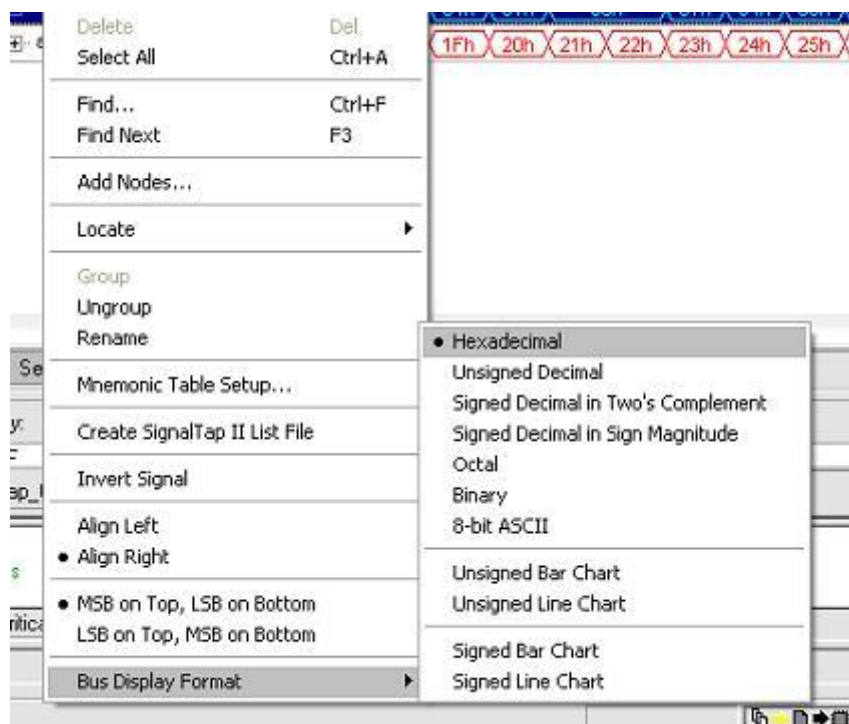
在 Instance Manager 中单击 按钮进行一次采样，



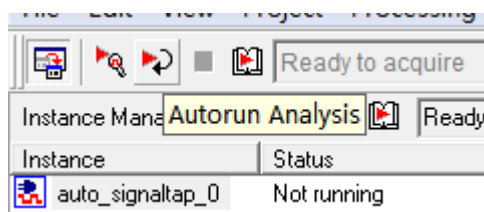
将会得到如图所示的波形。



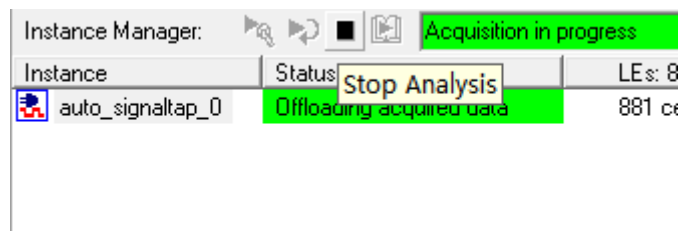
下面大家就可以看看看到的数据是否正确符合我们的设计意图啦，在信号名（比如 cnt 信号）上单击右键，选择 “Bus Display Format” 就可以选择显示方式，可以修改进制，甚至可以以模拟波形显示，大家可以修改一下看看效果



点击 Autorun Analysis，则采样会一直继续，此时观测窗口中的数据会一直改变。



点击 Stop Analysis 停止采样



## 2. 设置 SignalTap II 基本触发条件

在实际的工程调试中，我们不可能像上面那样漫无目的的查看数据，因为那样的话，SignalTap 采到的数据位置是不确定的，这样也就无法验证我们的设计是否与预期相符。所以下面我们就要告诉 SignalTap，在何时何种条件下来采我们感兴趣的数据。这样我们得到的数据才对我们调试时有意义的。

比如上面的程序：

VHDL：

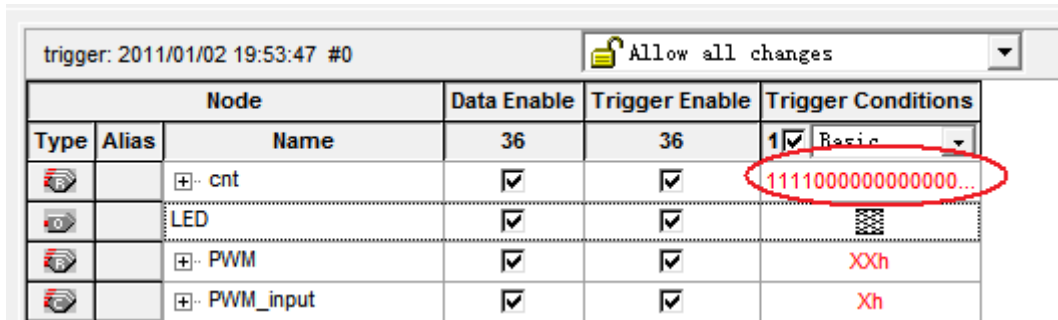
```
PWM_input <= cnt(24 downto 21) when cnt(25)='1' else not cnt(24 downto 21);--ramp the PWM input up and down
```

Verilog :

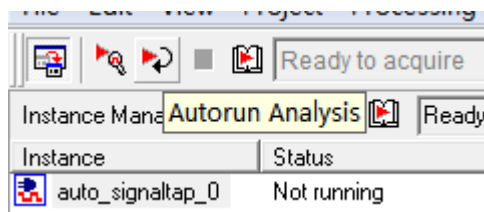
```
wire [3:0] PWM_input = cnt[25] ? cnt[24:21] : ~cnt[24:21]; // ramp the PWM input up and down
```

意思很简单，就是判断 cnt 的最高位，看看是不是 1，如果是 1 的话，就改变灯闪烁的分频比的趋势。但是这个功能究竟实现了吗，我们也无从知道，或者说，在某个时刻，如 cnt 为 11110000000...0 时，这两行代码是否是有效的，我们如何知道呢。这时候我们就可以利用触发抓住这时刻来看看是不是符合我们的设计意图。

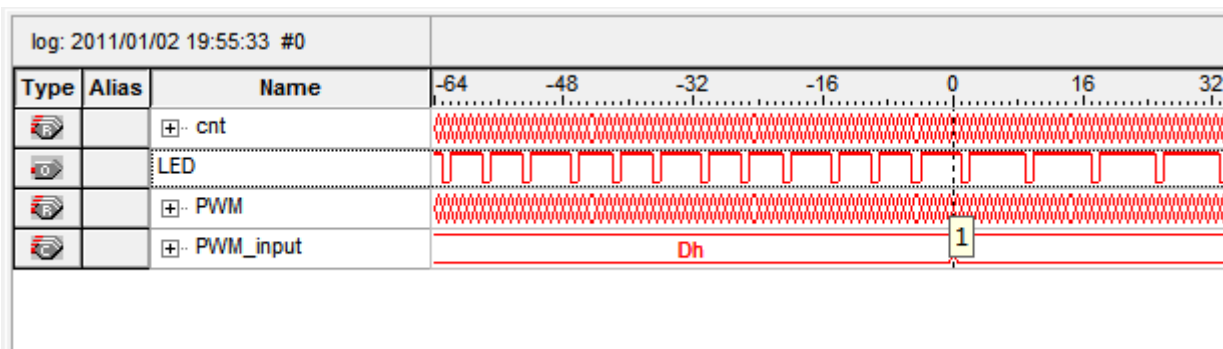
这时候我们回到 setup 选项卡，在 cnt 的 trigger conditions 条件里，输入 1111000000000000... 注意，这里改成 2 进制显示的，是 22 个 0，前面 4 个 1。别少了也别多了。



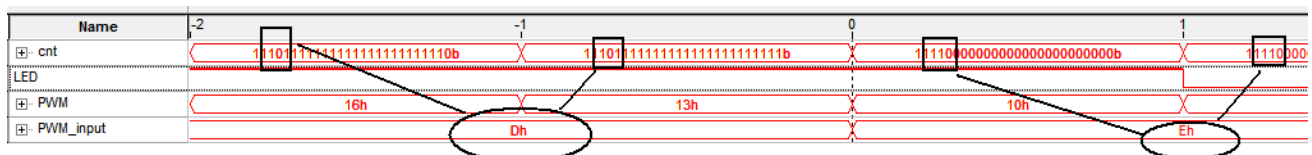
然后点击 Autorun Analysis，进行不断的触发采样








在如下的 0 位置附件，点击鼠标左右键，进行放大与缩小到适当的比例。









从下图可以看出，的确此时 PWM\_input 是按照前面的语句改变的



SignalTap 支持的基本触发条件除了由具体值来触发，也可以由电平，边缘等等触发，大家可以试一试。

Node			Data Enable	Trigger Enable	Trigger Conditions
Type	Alias	Name	36	36	1 <input checked="" type="checkbox"/> Basic
		+ cnt	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1111000000000000...
		LED	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
		+ PWM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XXh
		+ PWM_input	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Xh

 Don't Care  
 Low  
 Falling Edge  
 Rising Edge  
 High  
 Either Edge  
 Insert Value...





### 基本触发设置的注意事项

如果两个或者以上的信号都设置了触发条件，那么最终仅仅当这些条件同时满足时，采样才执行。他们之间默认是相与的关系。

## 3. 设置 SignalTap II 高级触发条件

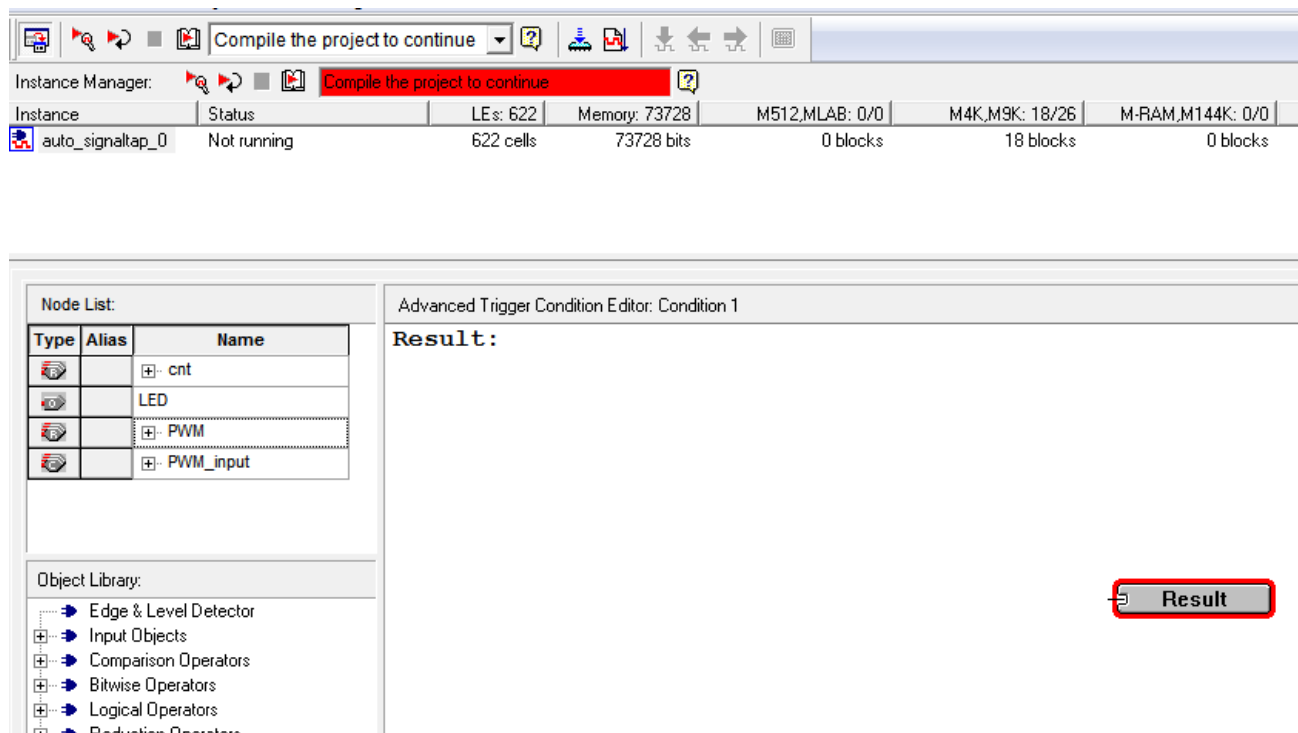
前面我们已经介绍了采用 signaltap 进行基本触发的设置方法，但我们知道这种触发条件有很多局限性，如多信号触发默认仅能够相与，只能够选择给定的数值触发，不能将条件设置为“如果不出现 XX 的时候触发”等等，所以当需要这样的特殊触发时，我们可以设置 SignalTap II 高级触发条件来满足我们的要求。

在如图所示的添加观察信号区域窗口中，将 Trigger Levels 选项改为 Advanced，将 Basic 改为 Advanced，则会弹出高级触发设置页面

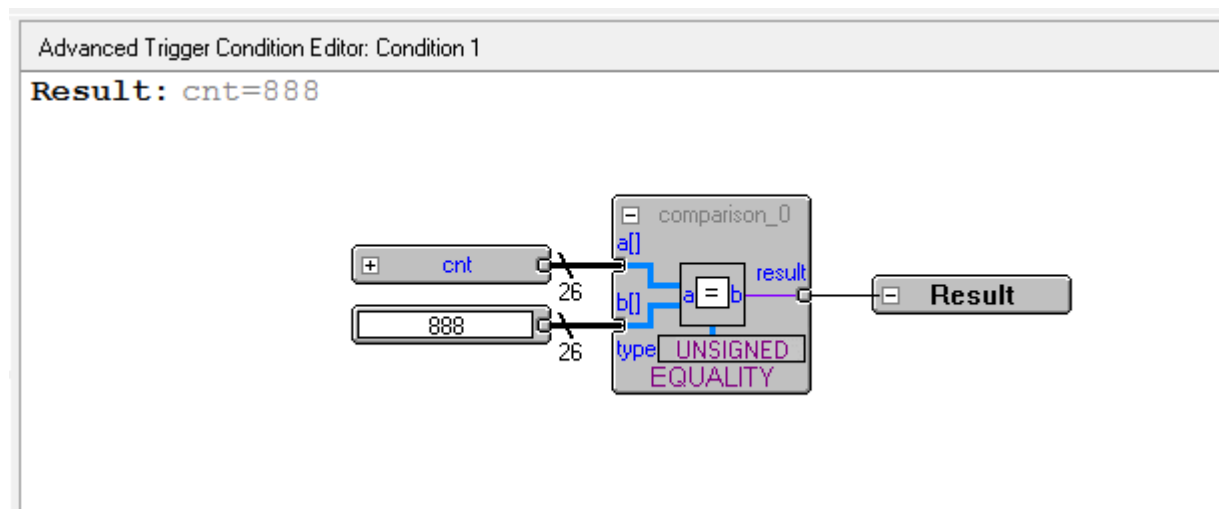
trigger: 2011/01/02 20:07:39 #0					Allow all changes
Node			Data Enable	Trigger Enable	Trigger Conditions
Type	Alias	Name	36	36	1 <input checked="" type="checkbox"/> Advanced
		+ cnt	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Basic Advanced
		LED	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
		+ PWM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
		+ PWM_input	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

打开 SignalTap II 文件高级触发设置

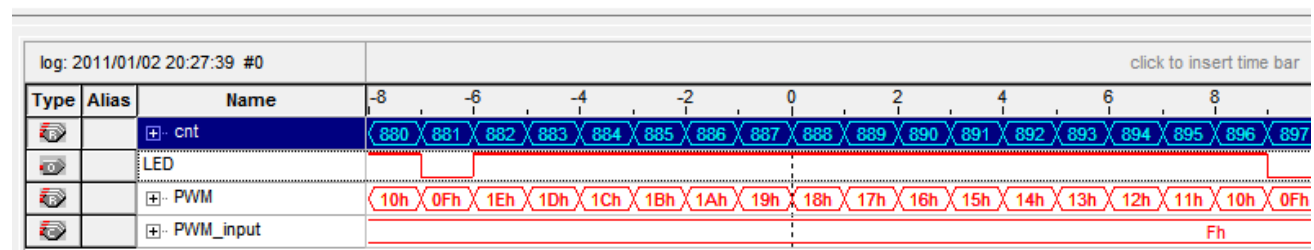
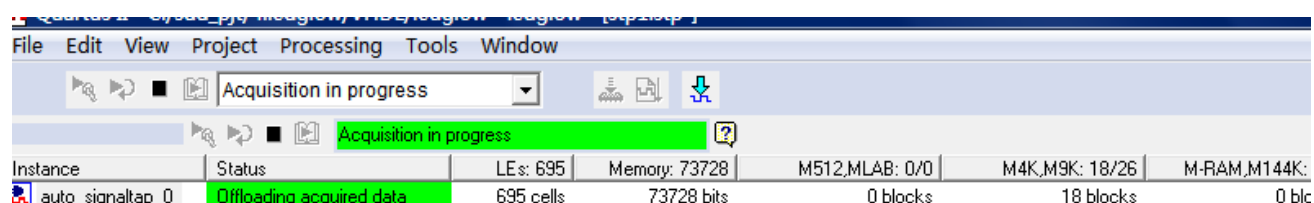




左下角的 Object Library 中有许多运算符，输入输出等。大家可以自行尝试一下。本例中只是作为一个简单的示意，设置为当信号 cnt = 888 时开始触发。从“node list”中拖入 cnt 信号，在“Object Library”/“Comparison Operator”中加入 equality，最后加入“Input Objects”/“Bus Value”，设置为 888。得到如图所示的触发条件。



改变高级触发条件，需要重新编译，编译完成后，重新下载，然后点击 Autorun Analysis。就可以看到最终的结果。



## 实验总结：

终于把signaltap的调试过程说完啦，虽然有些繁琐，但是大家以后就知道这是绝对值得的，因为signaltap的功能实在是太强大了，实际项目中是不可缺少的哦，但是不得不说的是，SignalTap是需要占用大量逻辑资源与存储区的，且quartus布局布线时必须将其与被测信号模块进行互联，这就难免会出现不稳定的情况，特别是程序较大，且时序要求较为严格时，有可能由于加入了signaltap导致实际的逻辑的布局布线受到影响，从而影响最终的功能。这些问题的处理与解决也是比较麻烦的，只能通过大家在今后的学习中通过实践多多体验吧。对了，别忘了看看板子上那可可爱的LED灯啊！

## 课后作业：

选择一个你感兴趣的例程，利用signaltap调试它。

## 相关信息

---

关于其他的相关信息，请访问以下网站

■ 购买本教程配套的开发套件，子卡或下载线缆：

<http://www.zr-tech.com>

■ 心得交流与问题互助：

<http://www.zr-tech.com/bbs>

## 版权信息

---

■ 本文档手册为ZRTech（[www.zr-tech.com](http://www.zr-tech.com)）原创资源，享有完全版权。

■ 任何收存和保管本文档各版本的单位和个人，未经本公司同意，不得随意复制、抄录、修改本文档的部分或者全部内容。

■ 转载本文档时请务必保证此文档的完整性。文档必须包含本版权信息。不得将转载作品以任何形式谋取商业利益，也不得向任何第三方提供，否则视为侵权。