AGM Supra 软件使用手册

Supra 软件系统需求

最小硬件要求：

- x86 64 位 Linux server
- x86 32 位或 64 位 Windows PC
- 内存 1G 以上
- CPU 1GHz 以上
- 硬盘空间 1GB 以上

操作系统需求：

- Linux kernel 2.9
- Windows XP，Windows Vista，Windows 7/8/10 (32 位/64 位)

预装：

Linux 或 Windows 系统，Supra 支持 AGM 自有 EDA 综合工具，和第三方 EDA 综合工具，包括 Synplify，Mentor 等。

亦可用 Altera Quartus II 的综合工具，需要安装 Altera Quartus II 8.0-13.0 版本（32 位/64 位），且 Quartus II 需要安装 Max II/V 以及 Cyclone II/IV 器件库。下面主要介绍 Supra 基于 Quartus II 综合的设计流程，其它工具设计流程类似。

1 版本命名规则

yyyy.mm.s# (hhhhhhhh)

yyyy.mm 分别是指年和月，s#是主次序列号。之前是 a0，现在升级为 b0。括号里的 hhhhhhhh 是内部开发版本 hash，用来区分同一个月内不同的 release。未来的主要升级将按照 alphabet 字母顺序，次要升级将使用数字递增，譬如 b0，b1，b2，c0，c1 等等。

2 软件下载安装

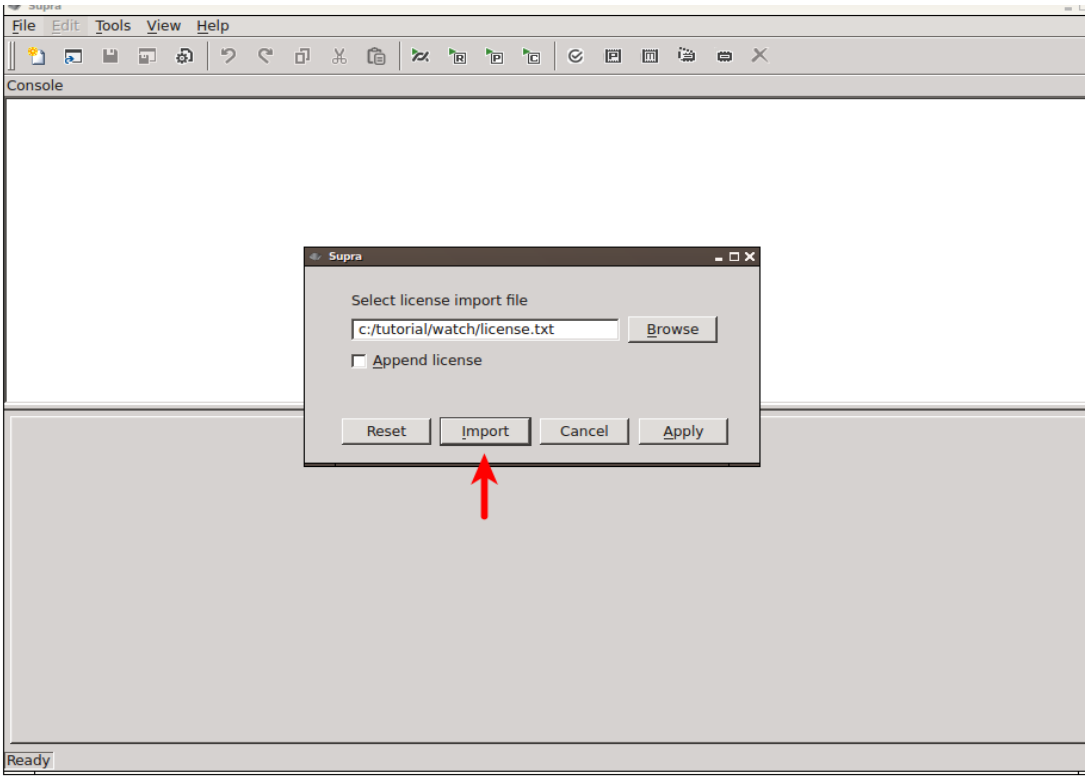
软件下载地址：链接 http://pan.baidu.com/s/1eQxc6XG 提取密码: q59e

选取所需系统版本下载，解压缩或执行安装。

注意：安装目录不能包含中文或空格等特殊字符。

- 运行 Bin 子目录下 Supra.exe。
- 导入 License。

选择菜单：“*File -> License -> Import license*”，选择 license 文件，点击“Import”按钮。如下图所示。



License 文件将被 copy 到 License 子目录，也可自行 copy 过去即可。

3 设计转换

对应器件型号列表：

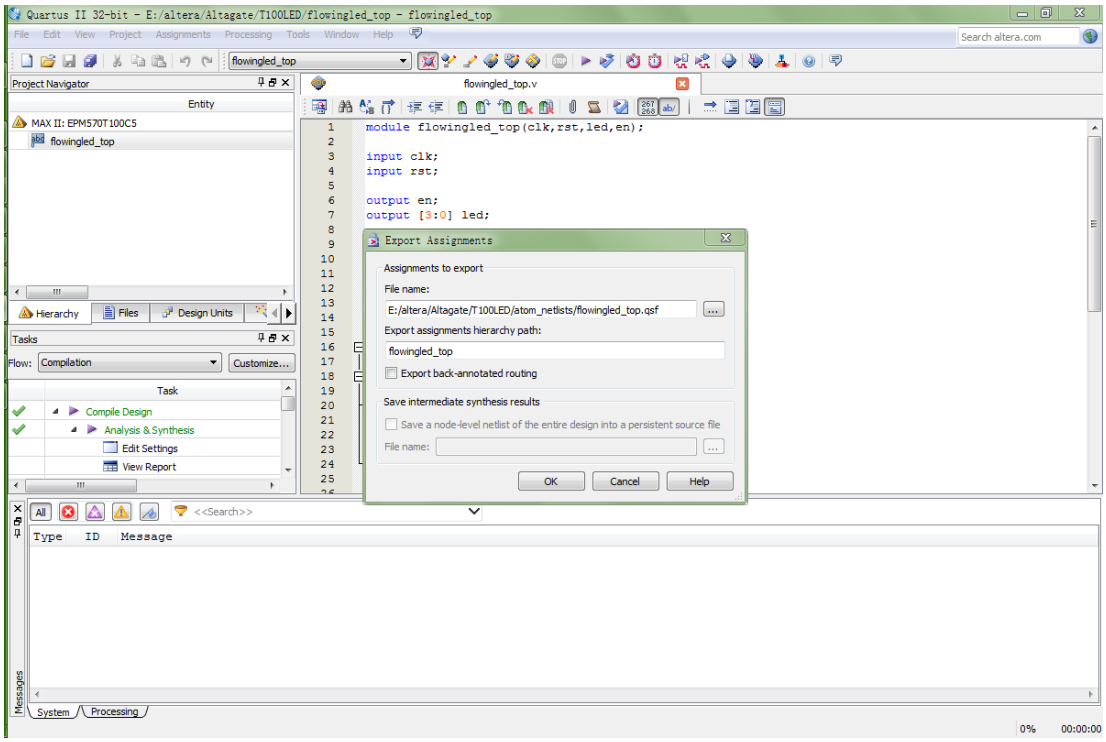
Altera 型号	AGM Supra 对应型号	封装
EPM240T100	AG256SL100	LQFP 100
EPM570T100	AG576SSL100	LQFP 100
EPM570T144	ALTA576S1T144	LQFP 144
EP4CE10E22	AG10KL144	LQFP 144
EP4CE10F17	AG10KF256	FBGA 256

3.1 准备原始工程<from_dir>

在 Quartus II 中完全编译成功的基于 MAX II 或 Cyclone IV 器件的工程目录。请确认选择与 AGM 器件管脚兼容的正确型号。

- 导出设置文件（AG10K/16K 系列不需要这一步）

在 QuartusII 中，选择菜单：“*Assignments -> Export Assignments...*”。使用默认选项，点击“OK”按钮。如下图所示。



一个<design>.qsf 文件将生成到<from_dir>/atom_netlists 目录下，用于项目转换。关闭此工程。

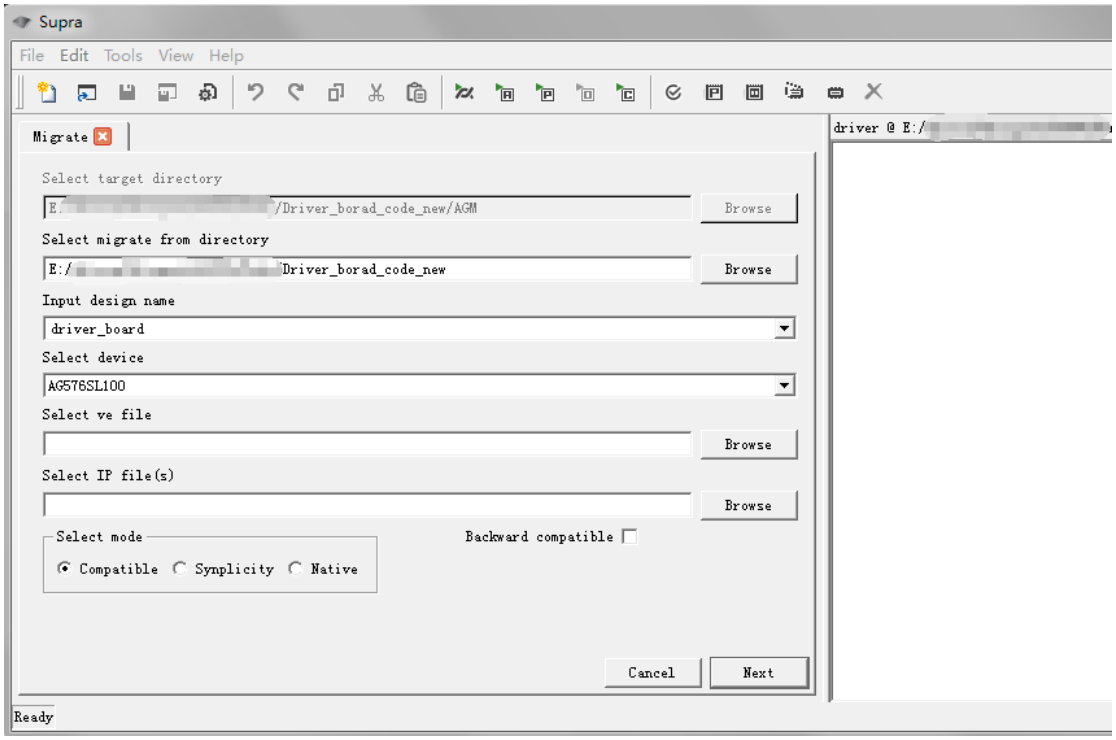
3.2 工程转换（Migrate）

打开 Supra，新建工程（File->Project->New Project），设置工程目录和工程名称。

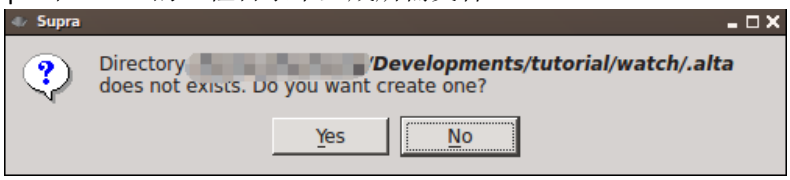
选择菜单：”*Tools -> Migrate*”，或者点击  按钮。

在 Migrate 界面填入信息：

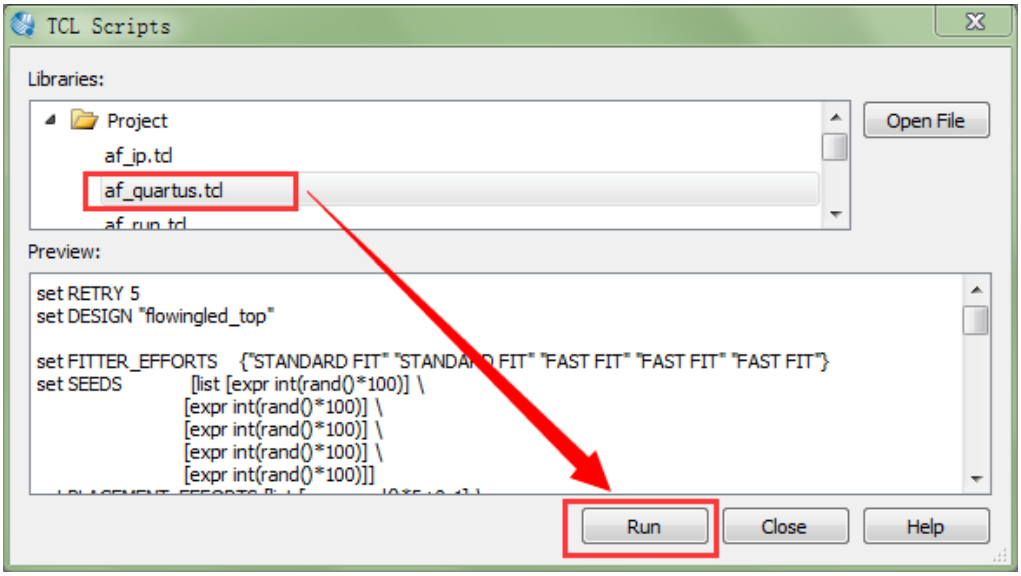
- 目标运行目录<run_dir>（AGM 工程目录）
- 原始项目目录<from_dir>
- 选择设计名称（应自动从<from_dir>中找到，请点选）
- 选择 AGM 器件型号
- 非管脚兼容器件需要添加 VE 文件，见相关文档
- 非兼容 IP，通过 Supra 产生 IP 文件后再添加
- Mode 选择 Compatible
- Backward Compatible 选项，如果使用老版本 Quartus II（无 Cyclone IV device）可以选上



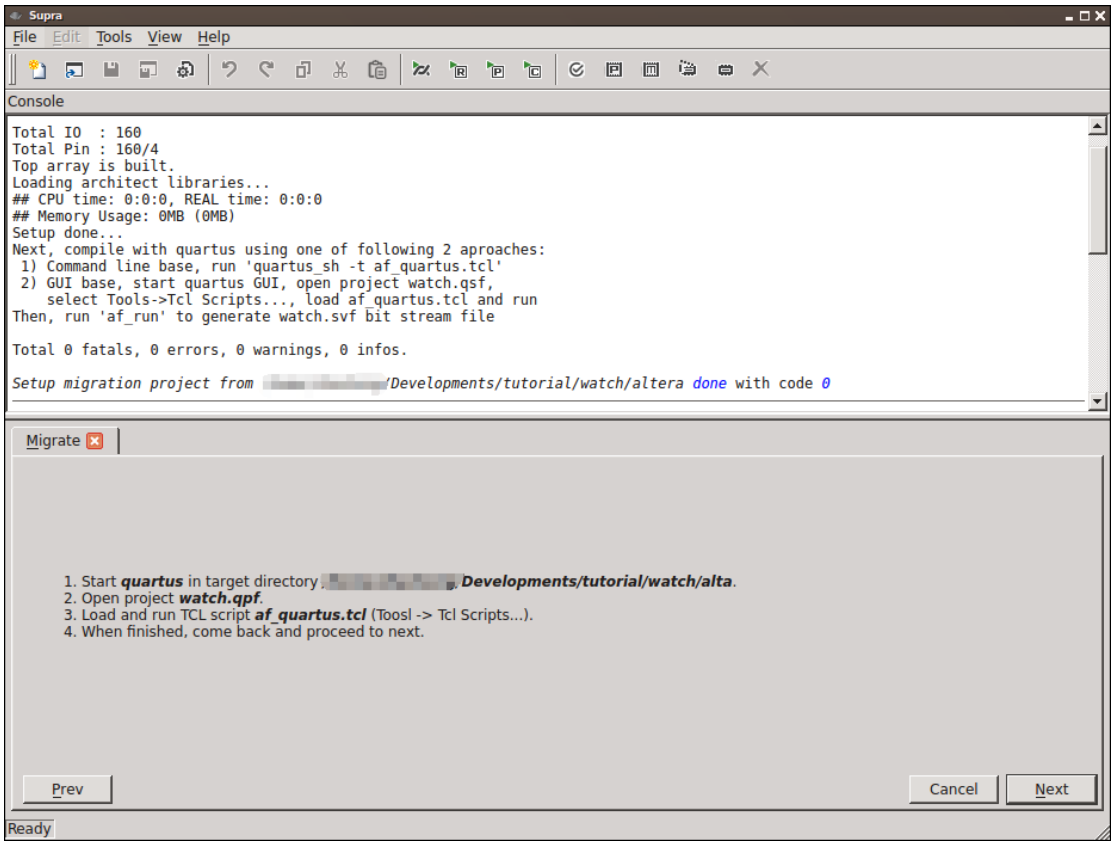
点击“Next”按钮。如开始未创建 AGM 的工程目录，可根据输入目录的名称自动产生，选择“Yes”。Supra 在 AGM 的工程目录中生成所需文件。



Quartus II 中打开<run_dir>中的<Design>.qpf 工程。选择菜单：“*Tools -> Tcl Scripts...*”，调用工程里的 af_quartus.tcl，运行点击“Run”按钮。

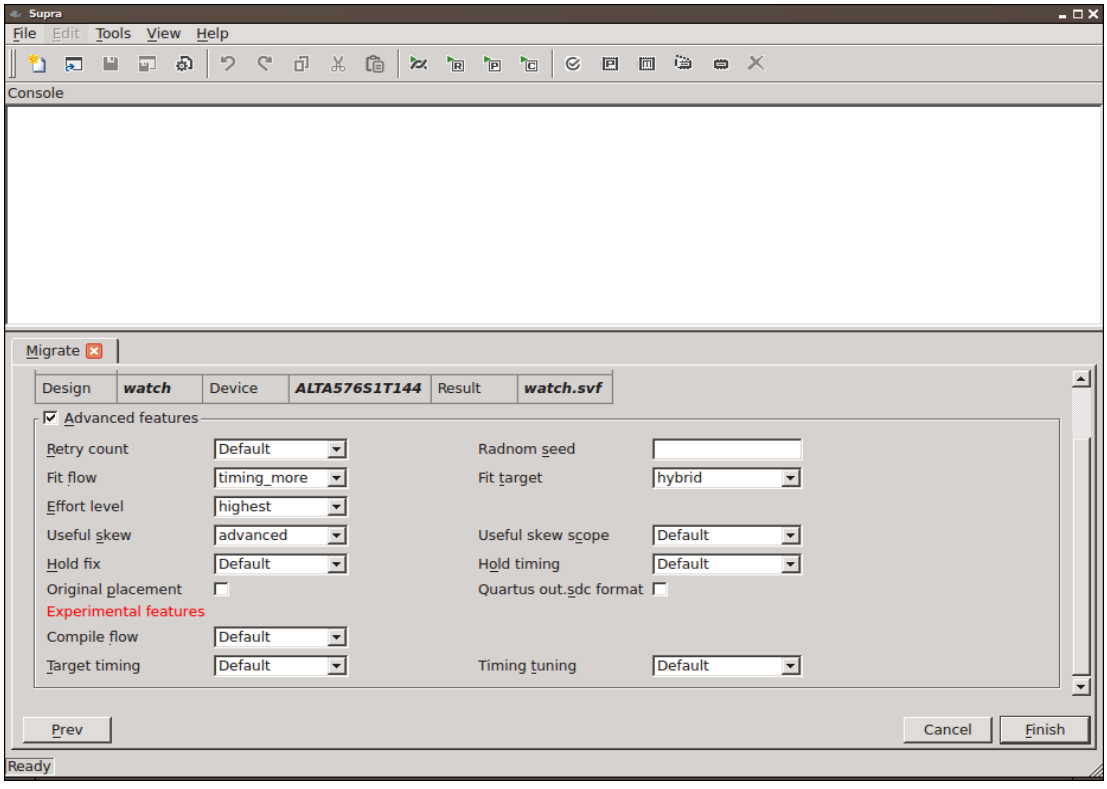


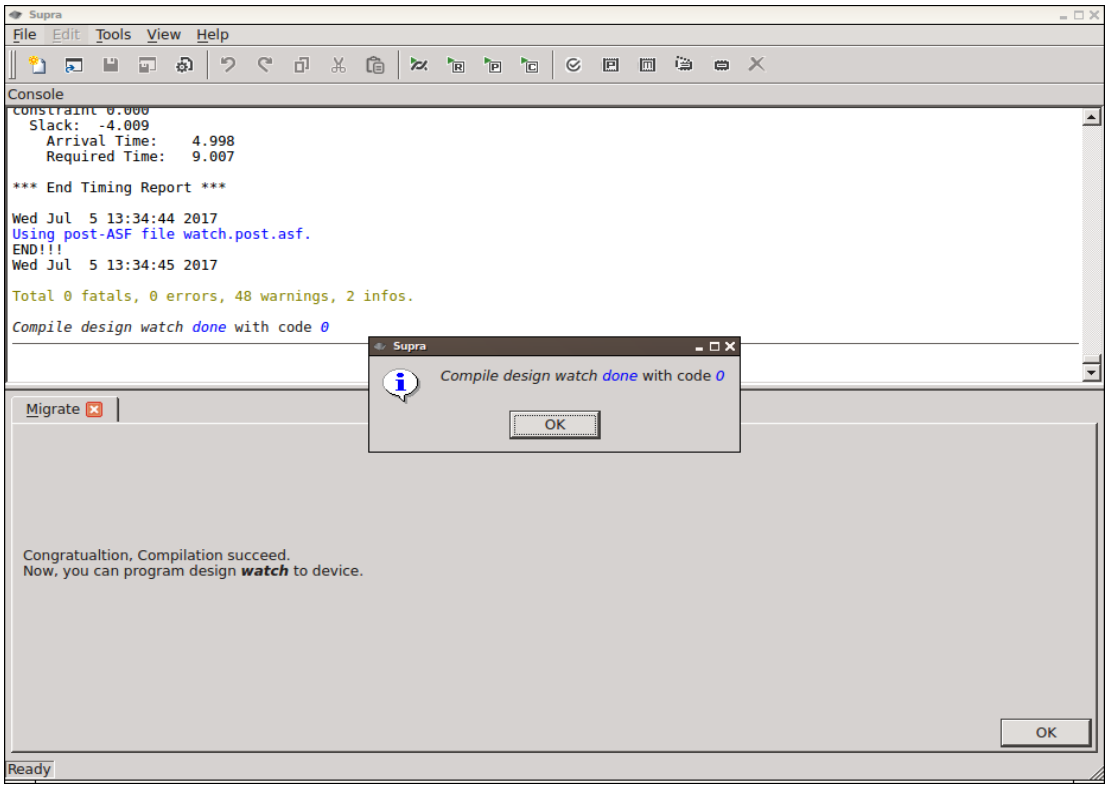
Quartus II 成功执行 Tcl 后，会综合出 Supra 需要的网表文件（vo）。退出工程，回到 Supra 软件。点击“Next”按钮。



3.3 Supra 工程编译

点击”Finish”按钮，Supra 开始编译工程，可在 Console 界面查看编译信息。



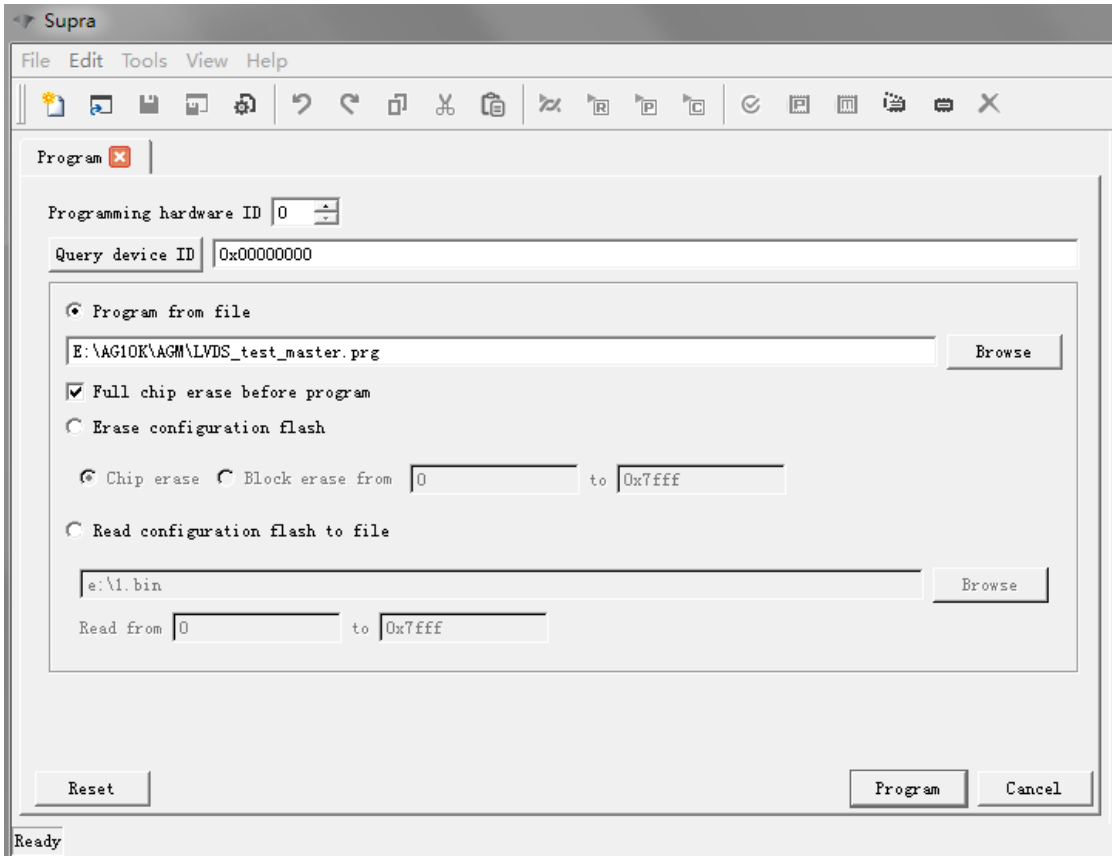


编译成功后，Supra 生成用于器件烧写的<design>.prg 等烧写文件。

转换和编译时产生的 log 文件，保存在工程的 alta_logs 目录下；编译的时序报告，保存在 alta_db 目录下，包括 setup 和 hold 时序的 rpt.gz 压缩文件中。

4 器件烧写

选择菜单：“*Tools -> Program*”，或者点击  按钮。



用 USB-BLASTER 连接器件 JTAG 接口，可以开始烧写文件。可通过 Programming hardware ID 选择本机上不同 USB-BLASTER，默认 ID 为 0，其它按 1，2，3 等顺序选择。

点击 Query 按钮，JTAG 正常连接器件即可显示 Device ID。

选择需烧写的 PRG 文件。（目前仅支持 Altera 标准下载线 USB-Blaster）。

点击“Program”按钮，开始通过 JTAG 烧写 PCB 上 AGM 器件。

- **烧写文件类型：**

AG256 / 272 / 576 系列：

<design> .prg 为烧写文件，通过 JTAG 烧写；

<design>_download.prg 为 AGM Downloader 专用编程器的烧写文件，用于批量快速烧写用；

<design>_SRAM.prg 文件为片内 SRAM 写入，通过 JTAG 烧写，掉电即失效，可用于设计调试；

AG10K/16K 系列：

<design>_SRAM.prg 文件为片内 SRAM 写入，通过 JTAG 烧写，掉电即失效，可用于设计调试；

<design>_master.prg 文件为 Master（AS）配置方式下，通过 JTAG 烧写外部配置 FLASH 的文件；

<design>_master.bin 为 Master（AS）配置方式下，配置 Flash 的标准烧写文件；<design>_master_as.prg 调用此 bin 文件，通过 AS 口直接烧写 FLASH；

<design>.bin 为基本配置文件，可用来产生其它类型烧写文件，也可用于 Slave（PS）配置方式；rbf 文件为此 bin 文件的字节高低位反向的转换后文件。

- **其它编程功能**

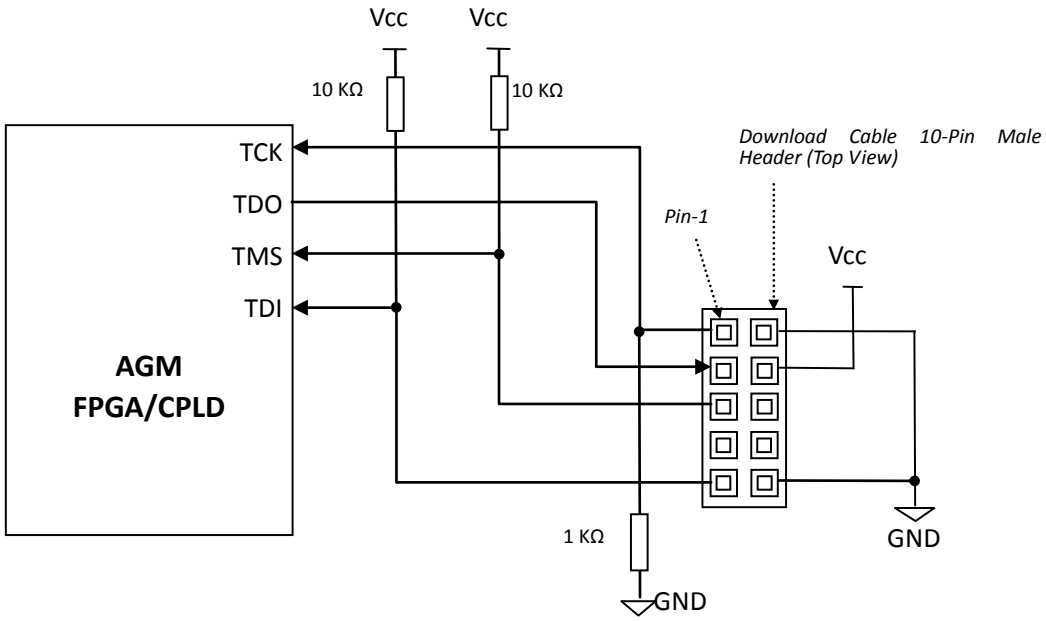
烧写<design>_master.prg 前需擦除 FLASH，可选择 Full chip erase before program。

通过 JTAG 擦除配置 FLASH，选择 Erase configuration flash 功能，可选全片擦除或擦除指定地址内容。

通过 JTAG 读取配置 FLASH 内容，选择 Read configuration flash to file，设置输出 bin 文件及读取地址。

5 JTAG 配置电路

AGM FPGA/CPLD 通过 JTAG 进行配置和烧写，利用 Altera USB-Blaster 下载线，硬件电路与 Altera 兼容，可参考下图：

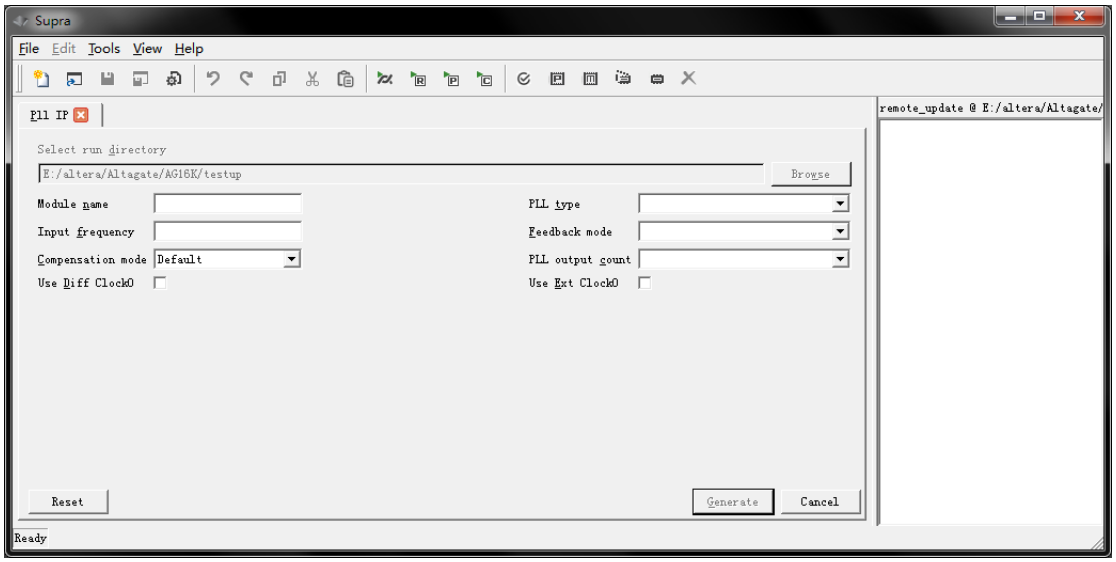


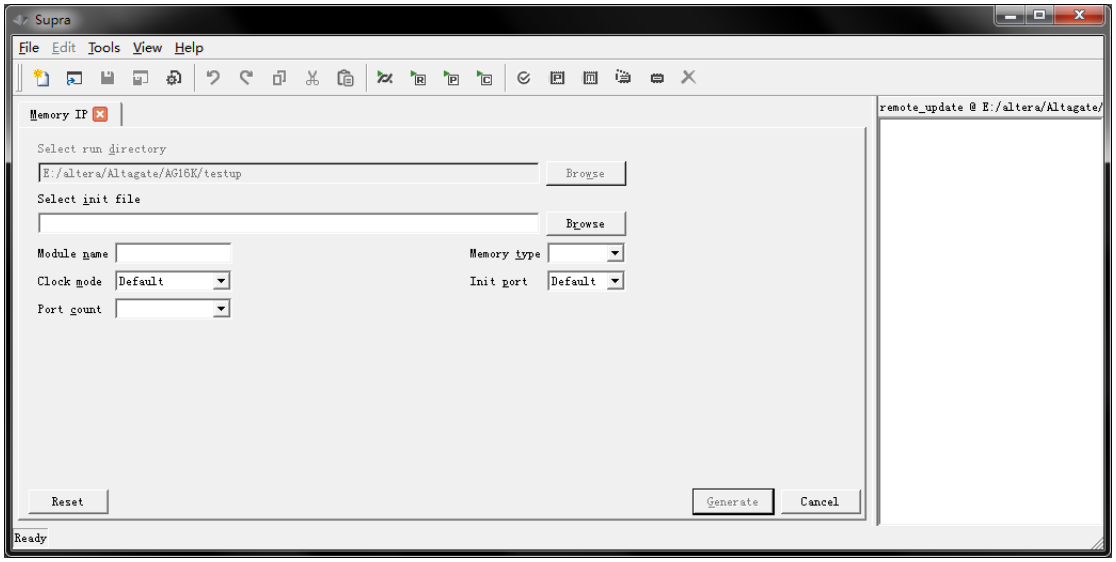
请注意，如果采用非标准 10 针插座，Pin-2 和 Pin-10 均需接地。

6 生成 IP

根据指定参数生成 PLL 以及 Memory IP 模组，此功能主要用于 AGM 专用非兼容 IP。具体应用请参考相关器件手册。

选择菜单：“*Tools -> Create IP -> Create Pll*”；“*Tools -> Create IP -> Create memory*”。



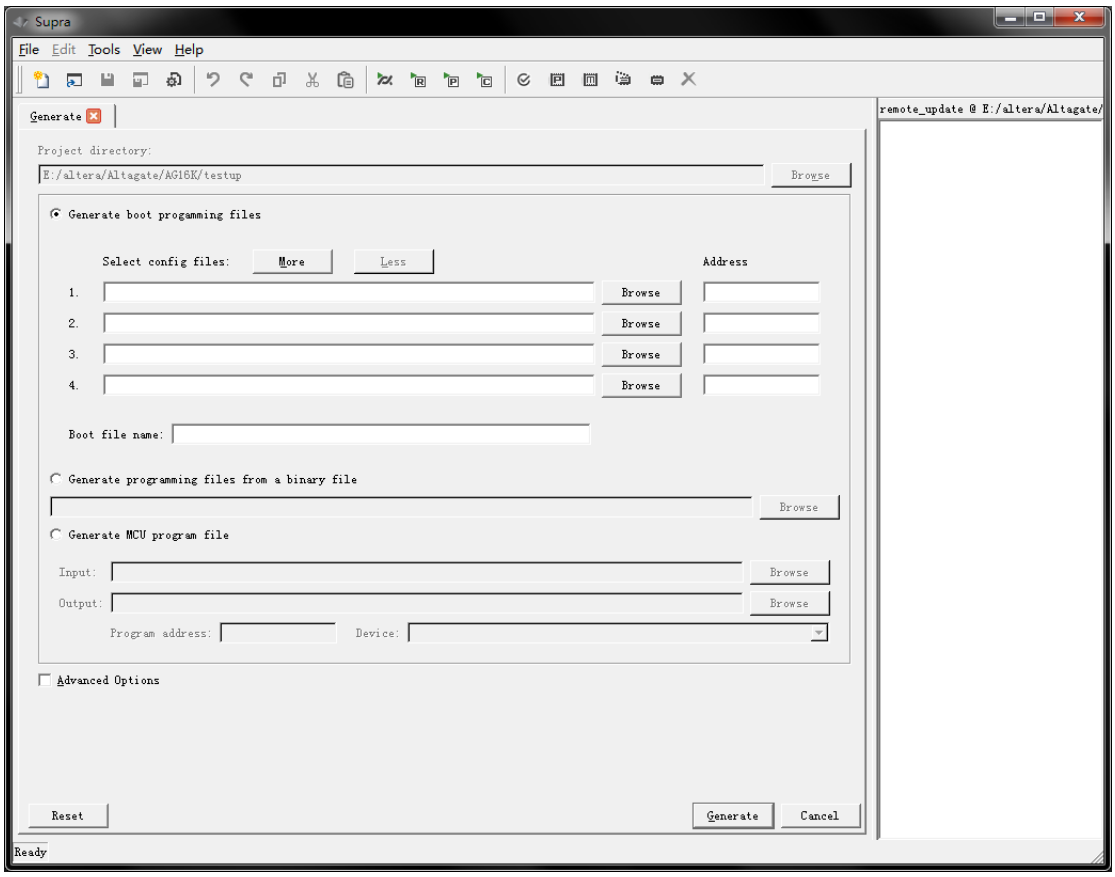


填写 PLL 或 Memory 参数。点击“Generate”按钮生成相关 IP 文件。

7 生成烧写文件

用于编程文件的后期加工生成。把 bin 文件转换成 JTAG 可直接烧写的 prg 文件。

选择菜单：“*Tools -> Generate*”。如下图所示：



AG10K/16K 系列支持多重启动，可以通过 Generate boot programming files 功能把多个设计文件组合成单一的烧写文件（选择每个设计文件的 design.bin）；也可以加入 SoC 器件中 MCU 的软件 bin 文件；Address 选择 FLASH 启动地址，16 进制格式，如：0x60000。设定输出文件名后，点击 Generate 会产生组合好的 bin 文件和烧写文件 prg。

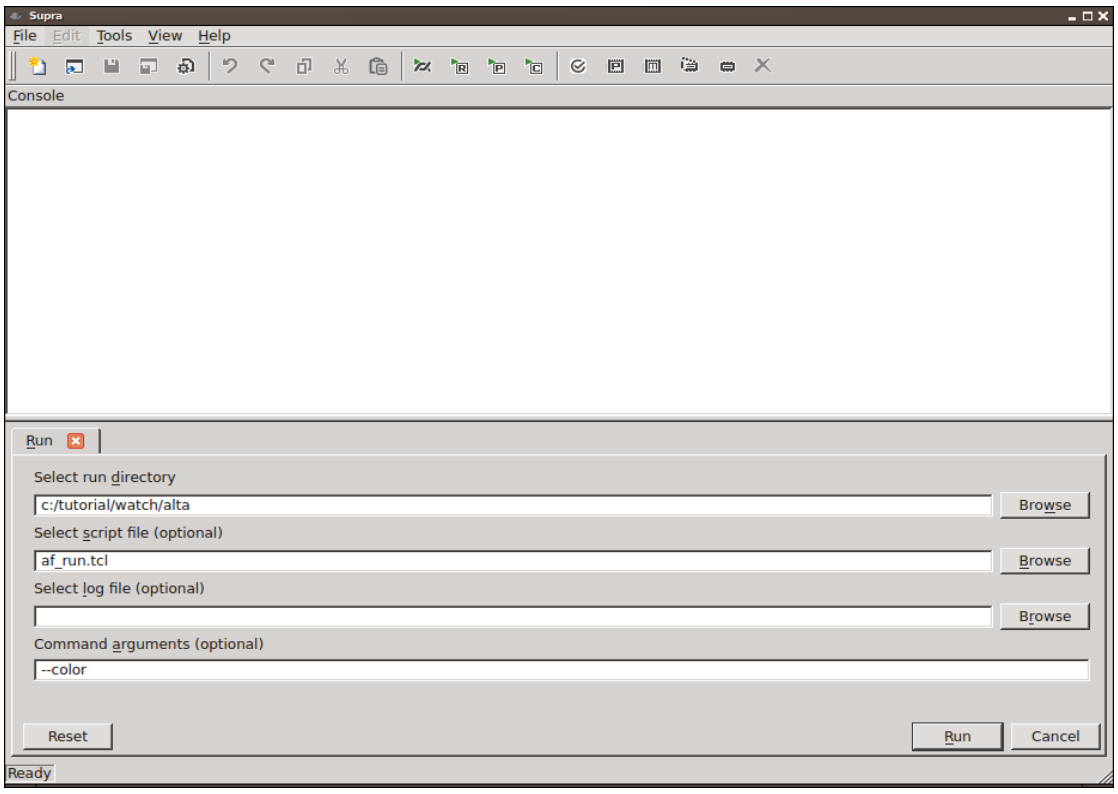
通过 Generate programming files from a binary file 功能，可以把用户自己的 bin 文件转换生成 prg 文件，通过 JTAG 直接烧写到配置 FLASH 中。

SoC 器件的 MCU 软件程序 bin 文件，需要加入 Header，可通过 Generate MCU program file 功能产生完整的 bin 文件。Input 编译软件（如 Keil）产生的 bin 文件，output 新的适用于 AGM SoC 的 bin 文件，需填写软件在 FLASH 上的启动地址，并选择相应器件型号。

8 执行 Tcl script

选择菜单：“*Tools -> Run script*”.

可以手动编写一份 Tcl script，通过 GUI 界面指定执行。如下图所示：



填写工作目录以及指定的 Tcl script 文件。点击“Run”按钮执行 script。在 Console 界面查看执行结果。

Supra 高级功能

1 其他功能

1）工程便捷编译

跳过 Migrate Setup,直接进入最后 Migrate Run 编译页面。

选择菜单：“*Tools->Compile*”。

2）工程编译高级功能

新增 Useful skew，Useful skew scope，Target timing 等，还有若干 Experimental 功能。请参考时序优化指南。

3）进程结束提示信息

选择菜单：“File -> Settings”中：

Elapse time：超过此时间后在编译完成会有提示信息和铃声。

Bell：提示音选择。

Popup message：弹出提示信息框。

4）Migrate 空的工程

Migrate 原工程目录（From directory）可以为空，这样在 Run 目录（Target directory）产生个空的工程；AG1KLP，AG3K 等非兼容型号可以这样开始设计。

5）Compile flow

Migrate 最后编译前的设置界面里，或者选择菜单：“*Tools -> Compile*”。

- Default/Full：Full place and route；
- Route：只运行 route，使用以前的 placement 结果；
- Probe：probe 内部信号；
- Skip：只运行时序分析，不重新 place and route。

选用"Route"模式，重新 route 可能会碰运气得到更好的布线结果；

选用"Probe"模式可用于 PCB 板的在线调试，可以通过 IO 快速探测内部信号。

"From"填需要探测的管脚或 net 名称，"To"是引出探测信号的 IO 管脚，可使用闲置 IO。如果这个 IO 是设计中使用的，需要选中"Force"，则强制引出到此 IO，否则就会编译出错。Probe 模式产生的编程文件和原设计的不同，会有"probe_"前缀。

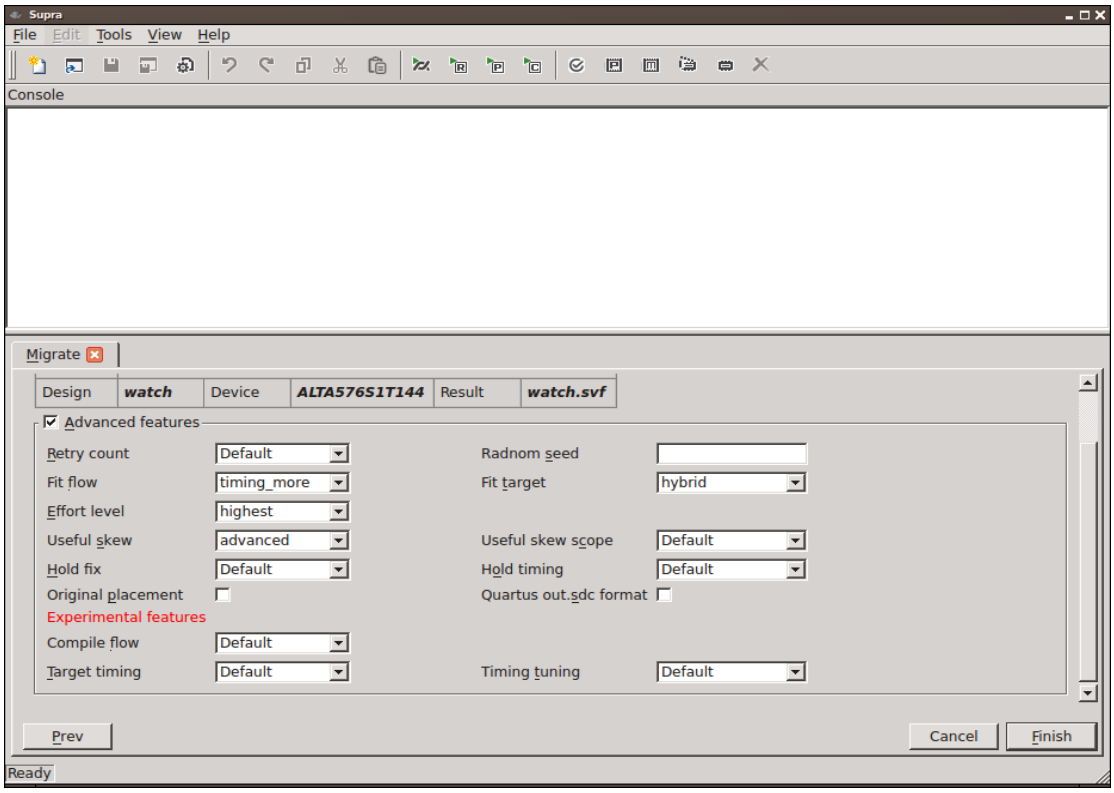
6）Batch Mode

可以通过随机 seed 及其它编译选项，自动批量多次编译，以便从中选择最优结果；根据需要选择编译次数和并行运行数量。批量编译结果保存在 bc_summary.txt 文件，以及 bc_results 目录中。

2 时序优化指南

Migrate Run 界面里提供了一些 Advanced Features 选项。这些选项可以针对不同的设计进行细分设定。第一次使用 Supra 的时候，会自动设定一组缺省设置。这些设置是综合以往的经验得到的最佳设定。如果通过这些缺省选项，不能得到满意的结果，则需要个别调整。

一旦一个选项被重新设置后，新的设置则会被记录下来。



针对时序优化，其中有几个选项最为重要。下面基于其先后顺序进行详细解说。

1）Fit flow

调整 routability 以及 timing 的 tradeoff 参数。推荐的默认设定是 timing_more，一般情况下会得到比较好的时序。其次是 timing 和 basic。

但如果设计比较拥挤，则可能有必要选取 routing 或者 routing_more。

2）Useful skew

通过调整时钟的 skew，从时序宽松的部分借来给时序紧张的部分，就像是劫富济贫。使用这个功能，对时序约束的要求比较高。如果欠缺一部分的约束，就可能把原本没有约束到但是其实时序紧张的部分反而拿来借给他处，造成额外的 violation。

鉴于上述的以及其他可能的副作用，要谨慎的使用该功能。应该先尝试不使用此功能。如果必须使用，需要从最基础的 basic 试起，其次是 advaned，最后再尝试 aggressive，boosted。随着设置 level 的增加，时序改善的可能性以及幅度都是相应增加，但是其副作用（如时序可能反而变坏）也可能会增加。

3）Useful skew scope

配合上述的 Useful skew 一起使用。指定 skew 影响的范围。all 是把 Useful skew 运用到所有的单元，core 是把 IO 输入输出部分排除，logic 是指运用在 slice 逻辑单元（排除运算以及存储单元）。由于通常 IO 上的约束不是很完善，所以有可能需要不影响到相关的时序。运算以及存储单元的结构跟逻辑单元相差比较大，有必要区分出来。

缺省的情况下，2）的 Useful skew 设定是 basic 的话，相当于 core，其他设定等同于 all。

如果编译结果报的时序没问题，但是实际上板子却出现时序问题，可以尝试逐渐选取其他设定。

4) Hold fix

保守的设计方式不太会产生大量的 hold 问题。一般 hold 问题主要出现在 IO 的 input/output delay 上。这个设定的缺省是只改进 IO 上的 hold 问题。但是如果有比较复杂的时钟，譬如不同时钟之间有时序关联，或者时钟上有不同的逻辑单元，或者有 local 时钟，则很有可能会产生内部的 hold 问题。这样就需要选择 basic 甚至 advanced 的了。主要这两个选项的对 fmax 的潜在影响会比较大。当发现有 hold 问题时，先排除 false path 的可能性，不要轻易启动这两档设定。

5）Target timing

指定优化目标的时序范围。由于发现有一些设计没有充分定义 false path 以及 multi-cycle path 约束，导致离现实相差很大的时序。编译算法会用很多资源去尝试改进没有必要改进的时序，反而会影响到其他正常的时序。可以通过这个选项把最坏的时序截到目标时序附近，进而排除掉一部分不现实的时序。这部分使用缺省的设定就可以了，偶尔必须时才需要调整设定。

6）Effort level

顾名思义，level 越高所花的编译时间越长，结果则可能会更好。

一般建议选 highest。

如果时序要求不高，可以选择其他更低的 level。

7）Random seed

编译算法有随机性，通过改变 seed，有可能会产生更好的结果。但是一般对结果的影响没有上述的几个设置大。

seed 0 有特殊含义，每次编译的 seed 都是随机产生的。

小于 100 的 seed，随机性要小于大于 100 的 seed。如果对编译结果的稳定性要求比较高的话，请选择小于 100 的 seed。

相反，大于 100 的 seed，结果的稳定性较差，但有可能得到更好的结果。

8) Fit target

一般建议选用 hybrid。

如果编译结果不理想，可以尝试选择其他选项。

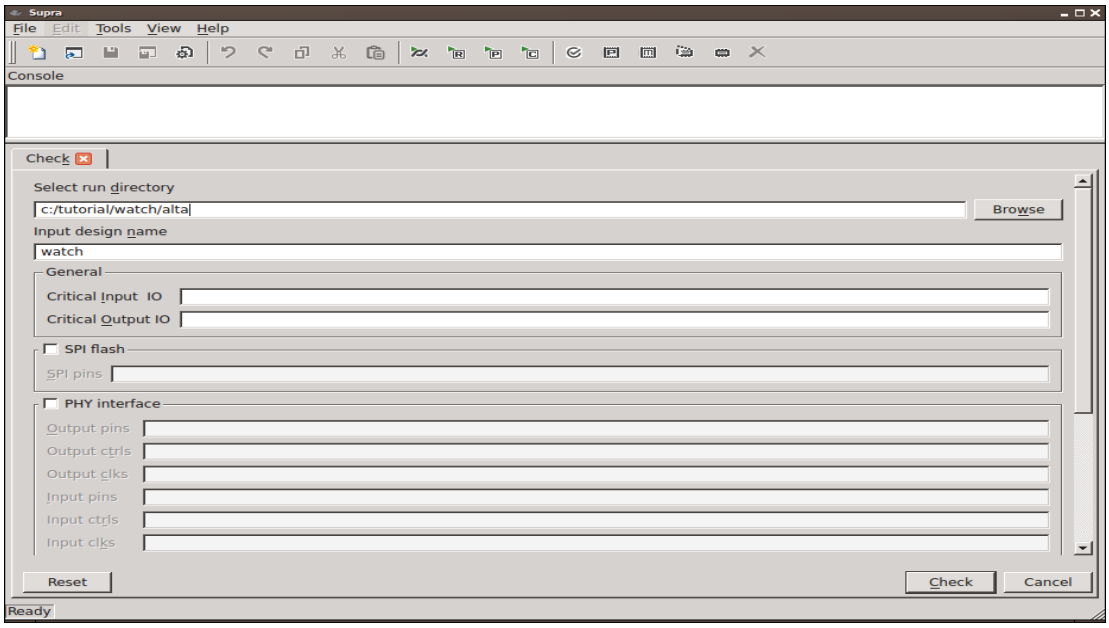
9）其他的设置

其他没有提到的设置，用缺省的设定就可以了。一般不需要改变设定。

3 设计规则检验

在当前工程目录下，产生一个<DESIGN>.chk 文件。在编译完成之后，就会自动进行设计规则检验。结果会放在 alta_db/check.rpt 文件里。

另外，也可以使用 GUI（*Tools->Setup rule check*）进行设置。如下图所示：



<DESIGN>.chk 文件的内容如下：

```
set SPI_SPI   "spi_sdi spi_sdo spi_sck spi_cs"
set PHY_OUT   "tx_data*"
set PHY_OCTRL "tx_ctl*"
set PHY_OCLK  "tx_clk*"
set PHY_IN    "rx_data*"
set PHY_ICTRL "rx_ctl*"
set PHY_ICLK  "rx_clk*"
set SDRAM_CHECK_METHOD 1
set SDRAM_DATA  "sdram_data[*]"
set SDRAM_ADDR  "sdram_add[*] sdram_ba[*]"
set SDRAM_CTRL  "sdram_ras sdram_cas sdram_we"
set SDRAM_CLK   "sdram_clk"
set SDRAM_CLOCK "sdr_clk_ext"
```

设计规则检验主要是针对 AG10K 芯片，包括以下几个模块。具体要根据设计需求，指定相关的参数。设计中无关的模块，可以省略掉。

- Critical Input IO/Critical Output IO

如果这些 Critical IO 没有使用 fast_input_register，fast_output_register，会有警告信息；

- SPI Flash

SPI_SPI (SPI pins) 指定所有（4 个）SPI 的端口

- PHY Interface

PHY_OUT （Output pins）指定 PHY TX 输出数据端口
PHY_OCTRL（Output ctrls）指定 PHY TX 输出控制端口
PHY_OCLK （Output clks）指定 PHY TX 输出时钟端口

PHY_IN （Input pins ） 指定 PHY RX 输入数据端口
PHY_ICTRL（Input ctrls） 指定 PHY RX 输入控制端口
PHY_ICLK （Input clks ） 指定 PHY RX 输入时钟端口

- SDRAM Interface

SDRAM_DATA (Data pins）指定 SDRAM 输入输出数据端口
SDRAM_ADDR (Address pins）指定 SDRAM 输出地址端口
SDRAM_CTRL (Control pins）指定 SDRAM 输出控制端口
SDRAM_CLK (Clock pins）指定 SDRAM 输出时钟端口
SDRAM_CLOCK (Output clock）指定时序约束里定义的 SDRAM 输出时钟
SDRAM_CHECK_METHOD (Method : recommend 1, recommend 2)

为了满足 SDRAM 输出端的时序，有两套不同的方法。

第一种办法，set SDRAM_CHECK_METHOD 1（Method : recommend 1）。跟输入端不同，输出端口不采用 FAST_OUTOUT_REGISTER，而是通过清零 SDRAM 时钟 PLL 上的相位来满足时序要求。

第二种办法，set SDRAM_CHECK_METHOD 2（Method : recommend 2）。跟输入端相同，输出端口采用 FAST_OUTOUT_REGISTER，然后通过调整 SDRAM 时钟 PLL 上的相位来满足时序要求。

- Clock skew

FPGA 的电路设计上的 clock skew 是很小的，基本上可以忽略不计。但是由于设计方式的原因，譬如时钟电路上有逻辑，或者没有用全局时钟，或者不同时钟之间有关联，都会产生不可忽略的 skew。进而对电路时序产生不良影响。

通过 Clock skew 检测，可以对较大的 skew 进行警告。对于这些警告，可以通过改进设计，完善时序约束（指定 false path 等）来消除。

- Clock cycle

有一些设计，有意无意的采用时钟上升下降沿的混搭。其结果，就是在某些路径上，实际用的是 half cycle。譬如 full cycle 是 8ns 的时钟，如果是 half cycle 的话，就会变成 4ns。4ns 对于一些稍复杂的路径，时序上会比较紧张。所以，建议在设计中应尽量避免 period 比较小的 half cycle。

通过 Clock cycle 检测，可以对 period 较小的 half cycle 进行警告。对于这些警告，可以通过改进设计，完善时序约束（指定 false path 等）来消除。