Application Report
ZHCA583 – November 2013

# 使用 CLT 工具优化 C6000 代码

Allen Yin
Communication Infrastructure

**摘 要**

在 C6000 DSP 的开发过程中，优化是必不可少的一个环节，根据对象不同可以分为系统，算法，代码以及内存优化。通常，开发者熟悉自己的代码，会从前三个方面修改以获得整体性能的提升，但是对于内存尤其是缓存（Cache）的优化，因为其涉及到芯片本身的架构，Cache 的维护由 DSP 自动完成，用户通常不能干预，所以似乎无从着手；考虑到这些实际的问题，从 TI 的 7.0 系列编译器开始支持使用缓存优化工具（Cache Layout Tools）对 C6000 代码进行优化，通过这一系列的工具，可以很轻松的完成 L1P Cache 性能的提升，本文详细介绍了该工具的使用方法。

## Contents



## Figures

# 1. 引言

目前，使用 TI DSP 的用户越来越多，在 C6000 系列 DSP 中，包含了 C64x, C64x+, C66x 等。在 C6000 DSP 的开发过程中，为了充分利用 DSP 的计算资源，需要对用户程序进行优化的工作，根据对象不同可以分为系统，算法，代码以及内存优化。通常，开发者熟悉自己的系统和代码，可以比较方便的从前三个方面修改以获得整体性能的提升，但是对于内存尤其是缓存（Cache）的优化，因为其涉及到芯片本身的架构，Cache 的维护由 DSP 自动完成，用户通常不能干预，所以似乎无从着手；考虑到这些实际的问题，从 TI 的 7.0 系列编译器开始支持使用缓存优化工具（Cache Layout Tools）对 C6000 代码进行优化，通过这一系列的工具，可以很轻松的完成 L1P Cache 性能的提升，本文详细介绍了该工具的使用方法。

# 2. C6000 DSP 内核缓存机制

C6000 系统的存储器结构如下图所示。

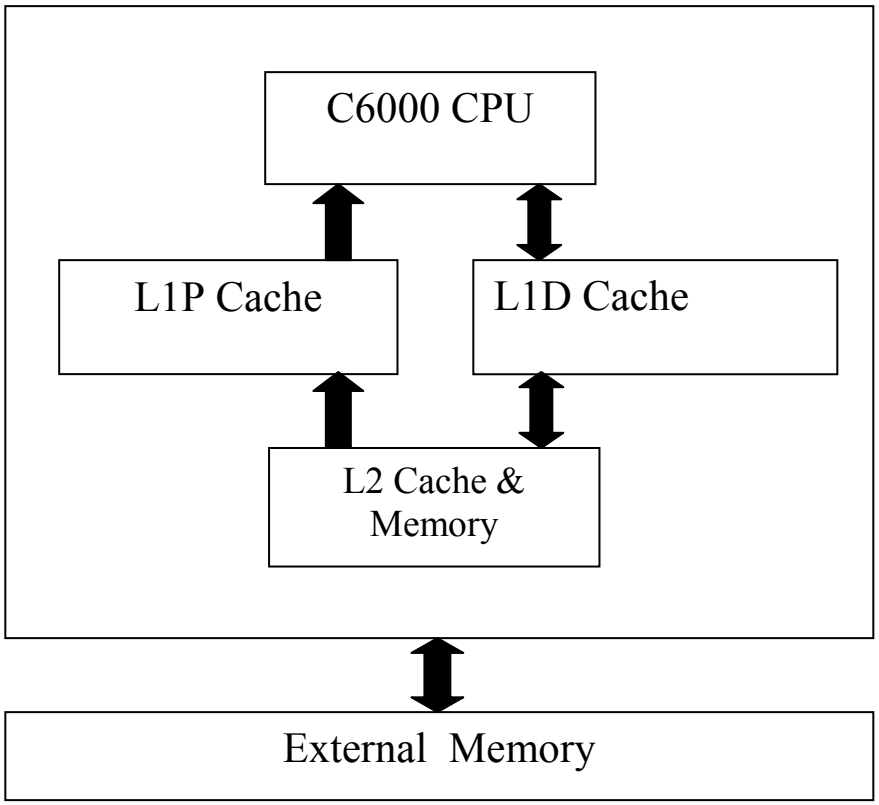



**Figure 1. C6000 存储器结构**

存储器分成三级：第一级是 L1，包括数据存储器（L1D）和代码存储器(L1P)；第二级是代码和数据共用存储器（L2 以及 MSMC SRAM）；第三级是外部存储器，主要是 DDR 存储器。L1P、L1D 和 L2 的 Cache 功能分别由相应的 L1P 控制器、L1D 控制器和 L2 控制器完成。

在 C6000 DSP 中通常我们会把 L1P 全部配置成 Cache，当 CPU 发出取指命令，首先会从 L1P 里查找，如果 L1P 找不到，则到下一级 Cache 或者 Memory 里查找，当找到需要的地址，则将其读入 L1P 里，CPU 从中读取执行。

因为 L1P Cache 的大小是有限的（本文以 32KB 为例），而用户内存空间一般大于 32KB， 必须采取一种映射的方式使得所有地址都能被 L1P 缓存；在 C6000 DSP 中，L1P Cache 使用地址直接映射，所有 DSP 核可访问的地址对 L1P Cache 大小（32K）取模就能得到该地址在 L1P Cache 的偏移值。

如果用户代码在内存排布不合理，可能会在 L1P Cache 中发生反复的内容替换，下图中的例子是一个极端情况。

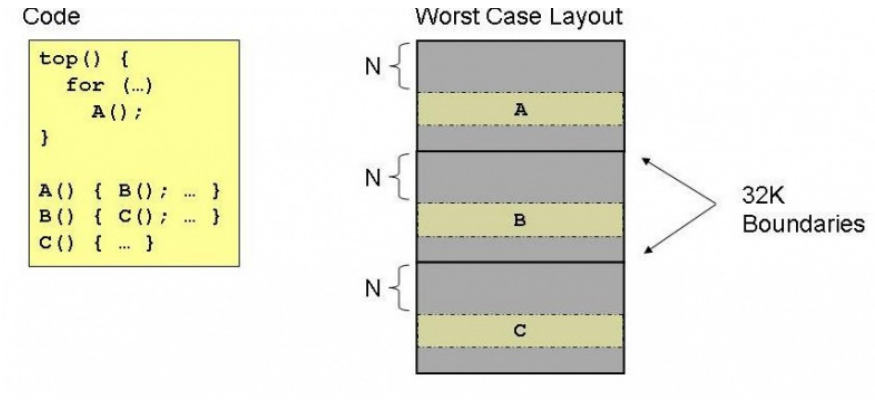



**Figure 2.　函数的不正确排布**

TOP 函数中 FOR 循环反复调用 A 函数，而 A,B,C 三个函数在内存地址的分布上，与 32KB 边界的偏移地址是一样的，因此，A,B,C 将对应 L1P 里同一个 CACHE 位置；其运行流程如下

- 当执行 A 时，CPU 需要把 A 函数调入到 Cache 偏移值 N 的位置上；
- A 调用 B，此时调入 B 到 Cache 偏移值 N 的位置上，覆盖 A 的代码；
- B 调用 C，此时调入 C 到 Cache 偏移值 N 的位置上，覆盖 B 的代码；
- C 返回，下一次循环调入 A 到 Cache 中覆盖 C 的代码。

DSP 核对 L1P，L2，DDR 的访问速度差异很大，对 L1P 的访问通常在 1 个时钟周期内完成，而 L2 平均需要 3-5 个周期，DDR 访问需要的时间更多，因此我们应该尽量避免上述这种反复重写 Cache 的情况，尽可能的减少函数在 Cache 中的置换。

如何解决该问题？最好的解决方法则是将 A, B, C 在内存中连续排放，这样对 Cache 的操作次数将降到最低，能够有效的提高执行效率，如下图所示，只要 A，B，C 总的大小不超过 32KB， 它们在 Cache 中的偏移值就是连续的，不会发生覆盖的现象，即使其总和大于 32KB，发生置换的也仅仅是超过 32K 的部分。

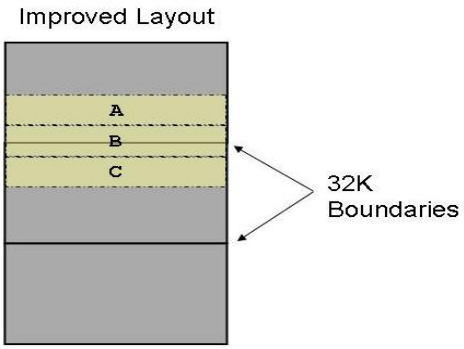



**Figure 3.　函数的正确排布**

## 3. 内存优化工具

通过上述机制可以看到，对于 L1P Cache 的优化主要通过分析函数调用关系和其在内存的分布。由于用户代码日益复杂，人工分析代码调用关系和地址排布需要花费大量的时间。因此，从 7.0 系列编译工具开始，TI 提供了一套内存优化工具 (Cache Layout Tools) 来帮助用户轻松快捷地解决该问题。

该工具的原理是在用户进行程序编译时打开生成分析信息选项，编译器会自动加入分析记录代码到用户程序里，之后用户在 TI DSP simulator 或者 DSP 芯片上运行该可执行文件，内置的分析代码会自动记录用户的函数调用关系及调用次数。运行的案例越多，记录的信息会更详细，优化的效果也就越好。

在得到函数运行时信息以后，就可以使用编译器工具对其进行分析，生成函数排布的顺序，最后将此排布顺序输入到编译器里重新编译原代码，生成的可执行文件就已经优化过内存排布，具体的操作可以参照以下实例。

## 4. 实例教程

该实例主要由三个 C 文件组成，

```
main.c:
defines main()
main() calls rare() once
main() calls main.c:local() 20 times
defines static local()
main.c:local() calls lots() 20 times
lots.c:
defines lots(); globally visible
lots() calls lots.c:local() 20 times
defines lots.c:local()
rare.c:
defines rare(); globally visible
```

实例中使用 DSP 计数器 TSCL 来统计 cycle 数，子函数放在 sub 目录下。

使用实例的步骤如下，

1. 编译代码

使用 TI 编译器对该实例进行编译，为了产生用于 profile 的信息，需要在编译时增加 --gen_profile_info 选项。如果使用命令还形式，命令行下运行 Compile.bat 文件，cl6x 的具体参数可以参考 spru186 和 spru187 两篇文档，一般可以在编译器的安装目录下找到他们，如 C:\Program Files (x86)\Texas Instruments\C6000 Code Generation Tools 7.3.9\doc。

```
cl6x -mv64+ --include_path="C:\Program Files (x86)\Texas Instruments\C6000 Code
Generation Tools 7.3.9/include" --include_path=".\inc" --gen_profile_info main.c
.\sub\lots.c .\sub\rare.c -z -llnk.cmd -i"C:\Program Files (x86)\Texas
Instruments\C6000 Code Generation Tools 7.3.9/lib" -l"rts64plus.lib" -o
CLTTutoirial.out -m CLTTutoirial.map
```

同时在目录下生成 OBJ 和 ASM 文件，这个和我们的实验关系不大，可以不用关注。

out 文件是一会需要下载到芯片里运行的可执行文件，而 map 文件用于帮助我们定位 profile 信息存放的内存地址。

如果用户使用 CCS 编译工具，则需要在 Build 的属性里指定 Feedback 选项，然后正常编译即可生成携带分析代码的可执行文件。

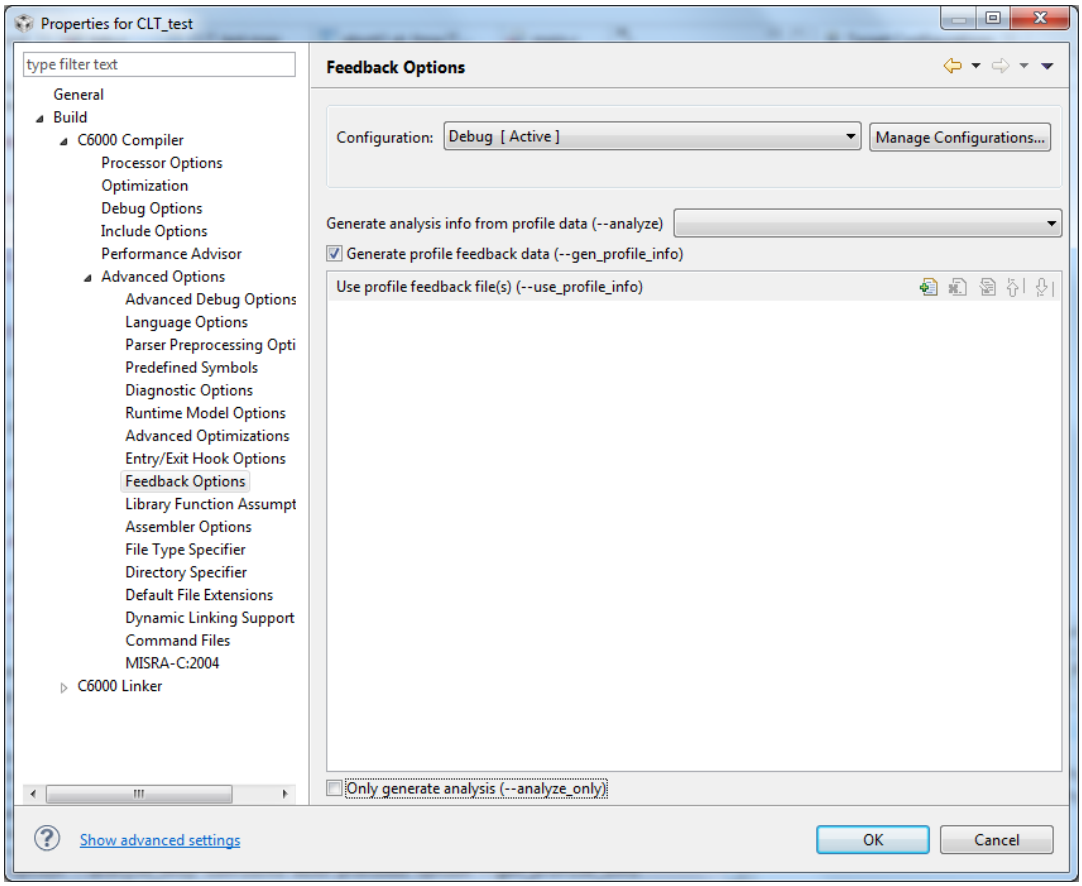



**Figure 4.　CCS 初编译的选项**

2. 获取分析信息

根据用户获取分析数据的不同，这里有两种方法，第一种方法适用于持续运行的程序，比如在基于 SYS/BIOS 的程序里，有些任务是以循环的方式存在的，这时用户需要自己从 DSP 内存里读取分析数据。

首先打开 map 文件，可以找到.ppdata 段的内存地址，这个地址就是 profile 信息存放处，在例子中

.ppdata 0 0081fecc 00000034 UNINITIALIZED

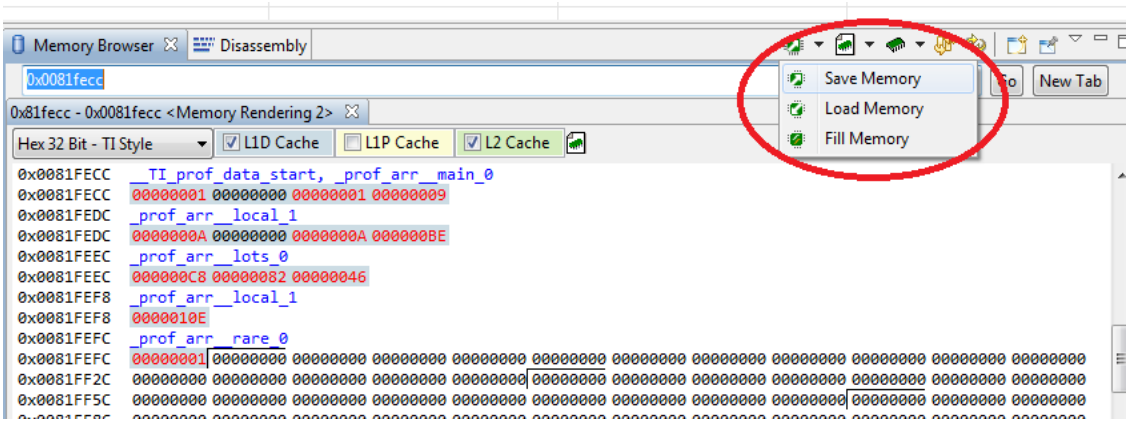
.ppdata 段位于 0x0081fecc 这个地址，长度是 34 个 byte。

启动 CCS，连接 EVM 板，下载 out 文件到 DSP 上，在 main 函数末尾加上调试断点，可以让程序到这里暂停（实际上，在用户代码中，可以把断点设置在需要的任何地方，profile 的信息是实时更新的）。

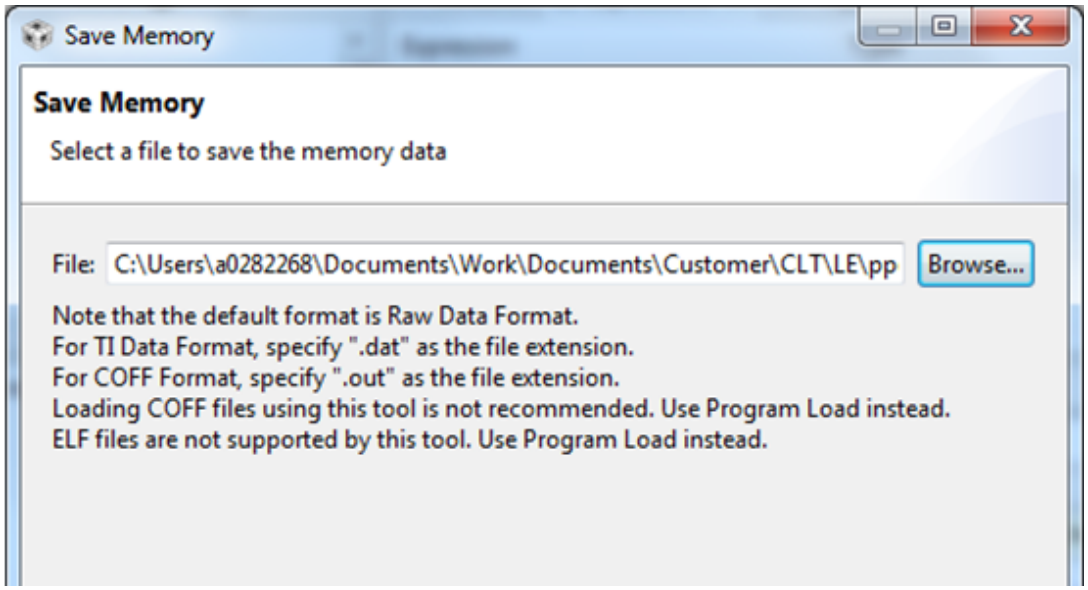
```
34
35      end = TSCL;
36      cycles = end-start-overload;
37      printf("The whole program need %d cycles!\n", cycles);
39  }
40
41  /*****************************************************************************/
42  /* LOCAL() - calls lots() 80 times.                                          */
43  /*****************************************************************************/
44  static void local()
45  {
46      int i;
47      printf("-");
```

运行该程序，到达断点后，在 View 菜单里打开 memory browser，将地址设定为 0x0081fecc, 可以读到.ppdata 的信息，参考以下步骤将其存到工程目录下。

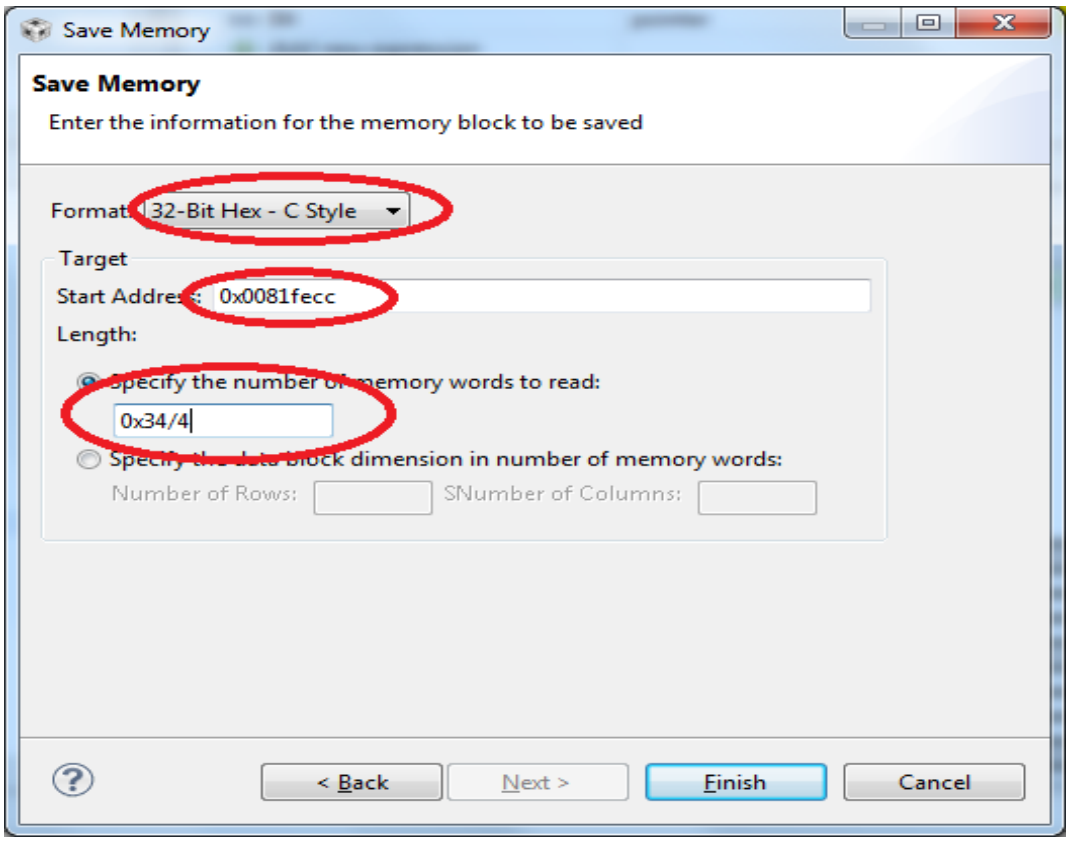
1) 选取 Save Memory




2) 存放路径




3) 确定数据地址和长度，如下图




4) 修改 dat 文件

打开刚才存下的 dat 文件，注意到文件头的数据长度是以 32 比特字为单位的，我们需要以 8 比特字节为单位，如

1651 9 81fecc 0 d 1

修改为

1651 9 81fecc 0 34 1

5) 转换文件格式

对刚才的运行 profile 信息进行分析，得到优化后的 cmd 内存排布文件，该文件内容如下，用户可根据自己的程序进行修改

```
Generate_pdatfile -le ppdata.dat pprofout.pdat
```

如果是大端，则将-le 选项改为-be 选项。

第二种方法，针对于只需运行一次流程的程序，CCS 可以自动生成 pdat 文件，需要注意的是，生成 pdat 文件的分析代码是在用户程序结束也就是 exit()程序执行时进行，因此用户要保证自己的程序能完整运行到主函数出口结束，否则无法生成 pdat 文件，需要用第一种方法来获取数据。

3. 重新编译代码

首先使用 pdd6x 从数据文件里提取 prf 文件作为重编译的输入文件

```
pdd6x pprofout.pdat -eCLTTutoirial.out -o=pprofout.prf
```

在命令行形式下，可以以以下形式调用输入文件生成 csv 文件，

```
cl6x --abi=coffabi -o2 -mv64+ --include_path="C:\Program Files (x86)\Texas
Instruments\C6000 Code Generation Tools 7.3.9/include" --use_profile_info=pprofout.prf
--analyze=callgraph  main.c .\sub\lots.c .\sub\rare.c
```

在 CCS 环境下，只需要在 CCS 里指定需要的数据文件后产生 csv 文件，

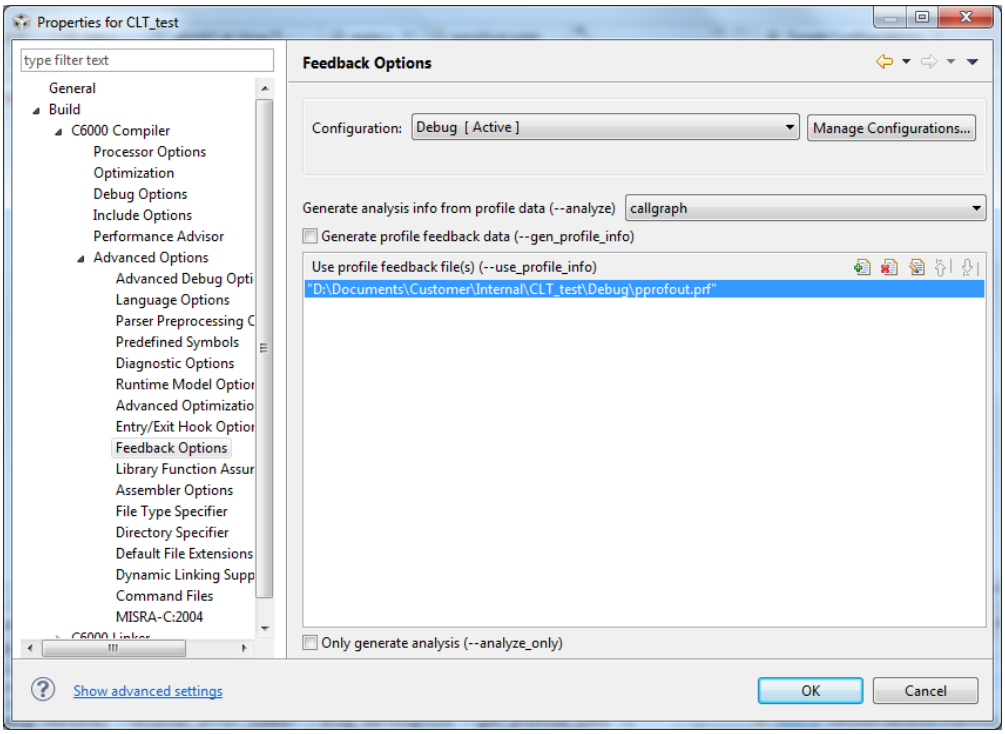



**Figure 5. CCS 重编译的选项**

通过调用 clt6x 生成内存排布

```
clt6x main.csv lots.csv rare.csv -o pfo.cmd
```

将输出的 pfo.cmd 加入到项目的 cmd 文件重新编译输出优化后的 out 文件，cache 优化到此完成。

对比优化结果，对于 TCP/IP 的例子应用上，CLT 带来了接近 20%的提升，对于视频编码等应用，CLT 也带来了 5%左右的提升。而且，用户代码量越大，则 CLT 可能带来的提升越明显。

# 4. 结论

通过使用 CLT 工具，可以方便快捷的对用户代码的 Cache 分配进行优化，用户不需要了解 DSP Cache 分配的详细信息，只需要在 Simulator 或者硬件板卡上运行定制的代码，就可以方便快捷地得到 Cache 的详细信息，并自动根据这些信息对程序在内存的分布进行配置已达到提升性能的效果。

# 参考文献

1. *TMS320C66x DSP CorePac User Guide* (SPRUGW0)
2. *KeyStone Architecture Multicore Shared Memory Controller (MSMC) User Guide (SPRUGW7)*
3. *KeyStone Architecture DDR3 Memory Controller User Guide (SPRUGV8)*
4. *Cache Layout Tools Example*
   *http://www.deyisupport.com/question_answer/dsp_arm/c6000_multicore/f/53/t/18908.aspx*

# 重要声明

德州仪器(TI) 及其下属子公司有权根据 JESD46 最新标准, 对所提供的产品和服务进行更正、修改、增强、改进或其它更改， 并有权根据 JESD48 最新标准中止提供任何产品和服务。客户在下订单前应获取最新的相关信息, 并验证这些信息是否完整且是最新的。所有产品的销售都遵循在订单确认时所提供的TI 销售条款与条件。

TI 保证其所销售的组件的性能符合产品销售时 TI 半导体产品销售条件与条款的适用规范。仅在 TI 保证的范围内，且 TI 认为 有必要时才会使用测试或其它质量控制技术。除非适用法律做出了硬性规定，否则没有必要对每种组件的所有参数进行测试。

TI 对应用帮助或客户产品设计不承担任何义务。客户应对其使用 TI 组件的产品和应用自行负责。为尽量减小与客户产品和应 用相关的风险，客户应提供充分的设计与操作安全措施。

TI 不对任何 TI 专利权、版权、屏蔽作品权或其它与使用了 TI 组件或服务的组合设备、机器或流程相关的 TI 知识产权中授予 的直接或隐含权限作出任何保证或解释。TI 所发布的与第三方产品或服务有关的信息，不能构成从 TI 获得使用这些产品或服 务的许可、授权、或认可。使用此类信息可能需要获得第三方的专利权或其它知识产权方面的许可，或是 TI 的专利权或其它 知识产权方面的许可。

对于 TI 的产品手册或数据表中 TI 信息的重要部分，仅在没有对内容进行任何篡改且带有相关授权、条件、限制和声明的情况 下才允许进行复制。TI 对此类篡改过的文件不承担任何责任或义务。复制第三方的信息可能需要服从额外的限制条件。

在转售 TI 组件或服务时，如果对该组件或服务参数的陈述与 TI 标明的参数相比存在差异或虚假成分，则会失去相关 TI 组件 或服务的所有明示或暗示授权，且这是不正当的、欺诈性商业行为。TI 对任何此类虚假陈述均不承担任何责任或义务。

客户认可并同意，尽管任何应用相关信息或支持仍可能由 TI 提供，但他们将独力负责满足与其产品及在其应用中使用 TI 产品 相关的所有法律、法规和安全相关要求。客户声明并同意，他们具备制定与实施安全措施所需的全部专业技术和知识，可预见 故障的危险后果、监测故障及其后果、降低有可能造成人身伤害的故障的发生机率并采取适当的补救措施。客户将全额赔偿因 在此类安全关键应用中使用任何 TI 组件而对 TI 及其代理造成的任何损失。

在某些场合中，为了推进安全相关应用有可能对 TI 组件进行特别的促销。TI 的目标是利用此类组件帮助客户设计和创立其特 有的可满足适用的功能安全性标准和要求的终端产品解决方案。尽管如此，此类组件仍然服从这些条款。

TI 组件未获得用于 FDA Class III（或类似的生命攸关医疗设备）的授权许可，除非各方授权官员已经达成了专门管控此类使 用的特别协议。

只有那些 TI 特别注明属于军用等级或“增强型塑料”的 TI 组件才是设计或专门用于军事/航空应用或环境的。购买者认可并同 意，对并非指定面向军事或航空航天用途的 TI 组件进行军事或航空航天方面的应用，其风险由客户单独承担，并且由客户独 力负责满足与此类使用相关的所有法律和法规要求。

TI 已明确指定符合 ISO/TS16949 要求的产品，这些产品主要用于汽车。在任何情况下，因使用非指定产品而无法达到 ISO/TS16949 要求，TI不承担任何责任。

| | 产品 | | 应用 |
|---|---|---|---|
| 数字音频 | www.ti.com.cn/audio | 通信与电信 | www.ti.com.cn/telecom |
| 放大器和线性器件 | www.ti.com.cn/amplifiers | 计算机及周边 | www.ti.com.cn/computer |
| 数据转换器 | www.ti.com.cn/dataconverters | 消费电子 | www.ti.com/consumer-apps |
| DLP® 产品 | www.dlp.com | 能源 | www.ti.com/energy |
| DSP - 数字信号处理器 | www.ti.com.cn/dsp | 工业应用 | www.ti.com.cn/industrial |
| 时钟和计时器 | www.ti.com.cn/clockandtimers | 医疗电子 | www.ti.com.cn/medical |
| 接口 | www.ti.com.cn/interface | 安防应用 | www.ti.com.cn/security |
| 逻辑 | www.ti.com.cn/logic | 汽车电子 | www.ti.com.cn/automotive |
| 电源管理 | www.ti.com.cn/power | 视频和影像 | www.ti.com.cn/video |
| 微控制器 (MCU) | www.ti.com.cn/microcontrollers | | |
| RFID 系统 | www.ti.com.cn/rfidsys | | |
| OMAP应用处理器 | www.ti.com/omap | | |
| 无线连通性 | www.ti.com.cn/wirelessconnectivity | 德州仪器在线技术支持社区 | www.deyisupport.com |

邮寄地址： 上海市浦东新区世纪大道 1568 号，中建大厦 32 楼 邮政编码： 200122