Intel C/C++、Fortran编译器的使用

李会民
hmli@ustc.edu.cn

中国科学技术大学 超级计算中心

2016-5-27

Intel C/C++、Fortran编译器简介

Intel C/C++ Fortran编译器是主要针对Inetl CPU平台的高性能编译器，在AMD Opteron平台上性能也不错，可用于开发复杂且要进行大量计算的程序。
Linux版本在x86_64（又称Intel64、EM64T或AMD64）平台上默认路径：

- 16系列：
  - */opt/intel/compilers_and_libraries_2016.0.109*：真正目录
  - */opt/intel/compilers_and_libraries_2016*：上述目录的链接
  - */opt/intel/compilers_and_libraries*：上述目录的链接
- 13-15系列：
  - */opt/intel/composer_xe_2015.1.133*：真正目录
  - */opt/intel/composer_xe_2015*：上述目录的链接
  - */opt/intel/composerxe*：上述目录的链接
- 12系列：
  - */opt/intel/composerxe-2011.3.174*
  - */opt/intel/composer_xe_2011_sp1.6.233*
- 11系列：*/opt/intel/Compiler/y/z*（如安装的为11.0.081，则对应的y为11.0，z为081）
- 10系列：C/C++编译器默认安装在*/opt/intel/cce/x*；Fortran编译器在*/opt/intel/fce/x*（x为版本号，如10.1.018）
- 用户可以查看*/opt/intel*类似目录下还安装有哪些版本

培训内容

编译命令基本格式

基本格式：

- C：*icc [options] file1 [file2 ...]*
- C++：*icpc [options] file1 [file2 ...]*

注意：

- *[ ]*表示是其内部的选项可选
- 文件名和选项区分大小写

串行及OpenMP并行程序编译举例

- 将C程序yourprog.c编译为可执行文件yourprog:
  icc -o yourprog yourprog.c
- 将C程序yourprog.c编译为目标文件yourprog.o:
  icc -c yourprog.c
- 将C程序yourprog.c链接*/opt/lib/liblapack.so*后编译为可执行文件yourprog:
  icc -o yourprog -L/opt/lib -llapack yourprog.c
- 将C程序yourprog.c静态编译为O3优化的可执行文件yourprog:
  icc -O3 -static -o yourprog yourprog.c
- 将C++程序yourprog.cpp编译为可执行文件yourprog:
  icpc -o yourprog yourprog.cpp
- 将OpenMP指令并行的C程序yourprog-omp.c编译为可执行文件yourprog-omp:
  icc -o yourprog-omp -openmp yourprog.c

编译时出错信息格式

```
netlog.c(140): error: identifier "hhh" is undefined
                        for(int hhh=domain_cnt+1;hhh>TMP;hhh--){
                                                 ^

netlog.c(156): error: expected an expression
    for(int i=0;i<32;i++)for(int j=0;j<256;j++)if(ip1[i][j]!=0)fprintf(fin,"202.38.%2d.%3d: %d\n",
        ^
```

- 源文件名(行数): 错误类型:具体说明
- 源代码，^指示出错位置

环境变量 I

- GCCROOT：gcc库路径，只有在因为添加-gcc-name选项导致找不到gcc库时才需要。
- GXX_INCLUDE：gcc头文件路径。
- GXX_ROOT：gcc库路径。
- LIBRARY_PATH：链接库路径
- LD_LIBRARY_PATH：链接共享库（.so库文件）路径
- OMP_*和KMP_*：OpenMP环境及其扩展变量
  - OMP：OMP_DYNAMIC、OMP_MAX_ACTIVE_LEVELS、OMP_NESTED、OMP_NUM_THREADS、OMP_PROC_BIND、OMP_SCHEDULE、OMP_STACKSIZE、OMP_THREAD_LIMIT、OMP_WAIT_POLICY

环境变量 II

- KMP：KMP_AFFINITY、KMP_ALL_THREADS、KMP_BLOCKTIME、KMP_CPUINFO_FILE、KMP_DETERMINISTIC_REDUCTIONS、KMP_DYNAMIC_MODE、KMP_INHERIT_FP_CONTROL、KMP_LIBRARY、KMP_MONITOR_STACKSIZE、KMP_SETTINGS、KMP_STACKSIZE、KMP_VERSION

- GNU和环境及其扩展变量
  - CPATH：C/C++头文件路径
  - C_INCLUDE_PATH：C头文件路径
  - CPLUS_INCLUDE_PATH：C++头文件路径
  - LIBRARY_PATH：库路径

编译过程

- 编译主要分为如下过程：
  - 预处理(Preprocessing)
  - 语义分析(Semantic parsing)
  - 优化(Optimization)
  - 代码生成(Code generation)
  - 链接(Linking)
- 前四项由编译器处理：icc或icpc
- 最后一项由编译器调用链接器处理：xild
- 编译时如添加了如下选项：
  - -c：编译器只生成目标代码（.o文件），需要再显式调用链接器生成可执行程序。
  - -E和-P：只生成预处理文件（.i文件）。
  - [Q]ipo：使用多文件过程间优化（又名全程序优化），将在链接时进行优化。
  - [Q]prof-gen：使用概要导向优化，将在链接时进行优化。

输入文件后缀与类型的关系

编译器默认将按照输入文件的后缀判断文件类型，编译时也可以用编译选项强制指定。

文件名	解释	编译时动作
filename.c	C源文件	传给编译器
filename.C filename.CC filename.cc filename.cpp filename.cxx	C++源文件	传给编译器
filename.a filename.so	库文件	传递给链接器
filename.i	预处理文件	传递给标准输出
filename.o	目标文件	传递给链接器
filename.s	汇编文件	传递给汇编器

输出文件后缀与类型的关系

编译器默认将输出按照文件类型与后缀相对应。

文件名	解释
filename.i	预处理文件，由-p选项生成
filename.o	目标文件，由-c选项生成
filename.s	汇编文件，由-s选项生成
a.out	默认生成的可执行文件

重要编译选项 I

编译选项对运行速度、兼容性等有影响，以下仅仅介绍部分重要选项，建议仔细看看编译器手册中关于程序优化的部分，特别是IPO、PGO和HLO部分，多加测试，选择适合自己程序的编译选项以提高性能。

- 输出及调试选项
  - -c：仅编译成目标文件（.o文件）
  - -debug [keyword]：启用或禁止生成调试信息。keyword可为none、full、all、minimal、extended、[no]parallel等
  - -g：包含调试信息
  - -g0：禁止产生符号调试信息
  - -o：指定生成的文件名。
- 预处理选项
  - -B*dir*：指定头文件、库文件和可执行文件的搜索路径
  - -Dname[=value]：指定传递给编译器的宏及宏值
  - -dD：输出源文件中的#define预处理指令
  - -dM：输出源文件中的宏定义。
  - -dN：与-dD类似，但只输出源文件中的宏名。

重要编译选项 II

- -gcc、-no-gcc和-gcc-sys：决定特定GNU宏（__GNUC__、__GNUC_MINOR__、__GNUC_PATCHLEVEL__）是否定义
- -icc、-no-icc：决定特定Intel宏（__INTEL_COMPILER）是否定义
- -U*name*：去掉特定宏的定义
- -undef：取消所有预定义宏
- -H：显示头文件顺序，并继续编译
- -I<头文件目录>：指明头文件的搜索路径
- -iprefix prefix：指定在头文件目录的先前搜索的路径
- -iquote dir：指定用""而不是<>引用的头文件的优先搜索路径
- -isystem*dir*：指定系统头文件的有限搜索路径
- -nostdinc++：不搜索C++的标准头文件路径，搜索其他标准目录
- -X：从搜索路径中去除标准库的路径

- 优化选项
  - -fast：最大化整个程序的速度，相当于：-ipo, -O3, -no-prec-div, -static, 和-xHost。这里是所谓的最大化，还是需结合程序本身使用合适的
  - -inline-level=[n]：设置inline层数

重要编译选项 III

- -ip：在单个文件中进行过程间优化(Interprocedural Optimizations-IPO)
- -ipo[n]：在多文件中进行过程间优化，n为可产生的目标文件数，为非负整数
- -mkl[=lib]：调用MKL中的特定库，lib可以为parallel（线程库，只是-mkl而没有=明确指定时，默认为此）、sequential（线性库）和cluster（集群特征库）
- -openmp：编译OpenMP程序，注意：只能在同一个节点的CPU上跑OpenMP程序
- -O<级别>：设定优化级别，默认为O2，O与O2相同，推荐使用。O3为在O2基础之上增加更激进的优化，比如包含循环和内存读取转换和预取等，但在有些情况下速度反而慢，建议在具有大量浮点计算和大数据处理的循环时的程序使用
- -Od：禁止所有优化
- -Ofast：设置某些激进参数优化程序速度，相当于：-O3 -no-prec-div
- -p：进行概要导向优化(Profile Guided Optimization-PGO)
- -simd和-no-simd：指定是否启用SIMD向量化

重要编译选项 IV

- -unroll[n]：循环最大可展开的层数，与性能相关
- -vec和-no-vec：指定是否启用向量化

- 代码生成选项
  - -ax*code*：生成针对Intel处理器的多重特征指定自动派发代码路径。code可以为MIC-AVX512、CORE-AVX512、CORE-AVX2、CORE-AVX-I、AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2等
  - -m*code*：指定为特征目标生成。code可以为MIC-AVX512、CORE-AVX512、CORE-AVX2、CORE-AVX-I、AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2等
  - -m32和-m64：指定生成为IA32或Intel 64架构的代码
  - -march=<CPU架构>：指定针对某种CPU架构的处理器进行优化。默认为pentium4，可以为generic、core-avx2、core-avx-i、corei7-avx、corei7、atom、core2、pentium*等
  - -mtune=<CPU架构>：指定针对某种CPU架构的处理器进行优化。默认为generic，可以为generic、core-avx2、core-avx-i、corei7-avx、corei7、atom、core2、pentium*等

重要编译选项 V

- -x*code*：指定针目标是何种处理特征。默认为generic，可以为CORE-AVX2、CORE-AVX-I、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2等

- 链接选项
  - -Bdynamic：在运行时动态链接所需要的库（.so文件）
  - -Bstatic：静态链接用户生成的库（.a文件）
  - -cxxlib：指定是否链接gcc提供的C++运行时库和头文件，-cxxlib[=dir]、-cxxlib-nostd和-no-cxxlib
  - -L<库目录>：指明库的搜索路径
  - -l<库文件>：指明需链接的库名，如库名为libxyz.a，则可用-lxyz指定
  - -pthread：使用POSIX线程库支持多线程库
  - -shared：产生共享目标而不是可执行文件，必须在编译每个目标文件时使用-fpic选项
  - -shared-intel：动态链接Intel库
  - -shared-libgcc：动态链接GNU libgcc库，可允许用户越过添加静态链接选项-static时的静态链接限制
  - -static：静态链接所有库

重要编译选项 VI

- -static-intel：静态链接Intel库
- -static-libgcc：动态链接GNU libgcc库，可允许用户越过添加动态链接选项-shared时的动态链接限制
- -static-libstdc++：动态链接GNU libstdc++标准库，可允许用户越过添加动态链接选项-shared时的动态链接限制
- -Wl,optlist：传递以,分割的链接参数给链接器
- -Xlinker val：传递以val变量（如链接参数、目标或库）直接给链接器

- 语言选项
  - -ansi：符合与gcc的ansi标准
  - -check [keyword[, keyword...]]和-nocheck：是否对某些条件进行检查，keyword可以为：none、[no]arg_temp_created、[no]assume、[no]bounds、[no]format、[no]output_conversion、[no]pointers、[no]stack、[no]uninit、all
  - -std=<标准>：标准可以为c89、c99、c9x、gnu89、gnu99、gnu++98、c++0x、c++11、gnu++0x，分别对应不同的标准。默认为-std=gnu89（C程序）和-std=gnu++98（C++程序）

重要编译选项 VII

- -stdlib[=keyword]：指定链接所用的C++库。可以为libstdc++（GNU libstdc++库）和libc++（libc++库）
- -strict-ansi：实现严格的ANSI兼容性
- -x <类型>：类型可为c、c++、c-header、cpp-output、c++-cpp-output、assembler、assembler-with-cpp或none，分别表示c源文件等，以使所有源文件都被认为是此类型的

- 数据选项
  - -align和-noalign：数据是否自然对齐
  - -fpic、-fPIC和-fno-pic：是否生成位置无关代码。当生成共享代码时，必须添加-fpic
  - -fpie、-fPIE：与-fpic类似，生成位置无关代码。不同之处在于-fpie生成的代码只能被链入执行程序
  - -mcmodel=*mem_model*：设定内存模型。mem_model可为：
    - small：限制代码和数据在开始的2GB地址空间，默认
    - medium：限制代码在开始的2GB空间，存储数据空间不受此限制
    - large：对于代码和数据存储空间都无限制

重要编译选项 VIII

- -check-pointers=keyword：是否检查使用指针访问内存时的数组边界。keyword可以为none、rw和write

- 其它选项
  - –help：显示帮助信息
  - -v：显示详细编译过程以及编译参数等
  - -V：显示编译器版本号
  - -w：编译时不显示任何警告，只显示错误
  - -wall：编译时显示所有警告

培训内容

支持Fortran标准

- Fortran标准：FORTRAN IV（即66）、77、90、95、2003和2008
- Intel Fortran编译器
  - 支持过时的和已经删除的Fortran特征
  - 完全支持Fortran 95(ANSI X3J3/96-007)和90(ANSI X3.198-1992)
  - 支持大多数Fortran 2003标准(ISO/IEC 1539-1:2004)
  - 开始支持Fortran 2008标准(ISO/IEC 1539-1:2010)的一些特征
  - 支持一些Fortran 2003标准的扩展（官方手册中用绿色表示）

编译命令基本格式

基本格式：

- Fortran： *ifort [options] file1 [file2 ...]*

注意：

- *[ ]*表示是其内部的选项可选
- 文件名和选项区分大小写

输入文件后缀与类型的关系

编译器默认将按照输入文件的后缀判断文件类型，编译时也可以用编译选项强制指定。

文件名	解释	动作
filename.a	目标库文件	传给编译器
filename.f filename.for filename.ftn filename.i	固定格式的Fortran源文件	被Fortran编译器编译
filename.fpp filename.FPP filename.F filename.FOR filename.FTN	固定格式的Fortran源文件	自动被Fortran编译器预处理后再被编译
filename.f90 filename.i90	自由格式的Fortran源文件	被Fortran编译器编译
filename.F90	自由格式的Fortran源文件	自动被Fortran编译器预处理后再被编译
filename.s	汇编文件	传递给汇编器
filename.so	库文件	传递给链接器
filename.o	目标文件	传递给链接器

输出文件的后缀与类型的关系

编译器默认将输出按照文件类型与后缀相对应。

文件名	解释	生成方式
filename.o	目标文件	编译时添加-c选项生成
filename.so	共享库文件	编译时指定为共享型，如添加-shared，并不含-c
filename.mod	模块文件	编译含有MODULE声明时的源文件生成
filename.s	汇编文件	编译时添加-S选项生成
a.out	默认生成的可执行文件	编译时没有指定-c时生成

环境变量 I

- GCCROOT：gcc库路径，只有在因为添加-gcc-name选项导致找不到gcc库时才需要
- GXX_INCLUDE：gcc头文件路径
- GXX_ROOT：gcc库路径
- LIBRARY_PATH：链接库路径
- LD_LIBRARY_PATH：链接共享库（.so库文件）路径
- OMP_*和KMP_*：OpenMP环境及其扩展变量
  - OMP：OMP_DYNAMIC、OMP_MAX_ACTIVE_LEVELS、OMP_NESTED、OMP_NUM_THREADS、OMP_PROC_BIND、OMP_SCHEDULE、OMP_STACKSIZE、OMP_THREAD_LIMIT、OMP_WAIT_POLICY

环境变量 II

- KMP：KMP_AFFINITY、KMP_ALL_THREADS、KMP_BLOCKTIME、KMP_CPUINFO_FILE、KMP_DETERMINISTIC_REDUCTIONS、KMP_DYNAMIC_MODE、KMP_INHERIT_FP_CONTROL、KMP_LIBRARY、KMP_MONITOR_STACKSIZE、KMP_SETTINGS、KMP_STACKSIZE、KMP_VERSION

- GNU和环境及其扩展变量
  - CPATH：头文件和module文件路径
  - LIBRARY_PATH：链接库路径
- 编译器运行时环境变量
  - F_UFMTENDIAN：小端到大端（little-endian-to-big-endian）数据转换时的文件号
  - FORT_CONVERTn：指定需要进行小端大端文件转换的文件号

重要编译选项 I

编译选项对运行速度、兼容性等有影响，以下仅仅介绍部分重要选项，详细的建议仔细看看编译器手册中关于程序优化的部分，特别是IPO、PGO和HLO部分，多加测试，选择适合自己程序的编译选项以提高性能。

- 输出及调试选项
  - -c：仅编译成目标文件（.o文件）
  - -debug [keyword]：启用或禁止生成调试信息。keyword可为none、full、all、minimal、extended、[no]parallel等
  - -g：包含调试信息
  - -g0：禁止产生符号调试信息
  - -o：指定生成的文件名
- 预处理选项
  - -B*dir*：指定头文件、库文件和可执行文件的搜索路径
  - -Dname[=value]：指定传递给编译器的宏及宏值
  - -dD：输出源文件中的#define预处理指令
  - -dM：输出源文件中的宏定义

重要编译选项 II

- -dN：与-dD类似，但只输出源文件中的宏名
- -gcc、-no-gcc和-gcc-sys：决定特定GNU宏（__GNUC__、__GNUC_MINOR__、__GNUC_PATCHLEVEL__）是否定义
- -icc、-no-icc：决定特定Intel宏（__INTEL_COMPILER）是否定义
- -fpp和-nofpp：是否对源代码进行预处理
- -U*name*：去掉特定宏的定义
- -undef：取消所有预定义宏
- -H：显示头文件顺序，并继续编译
- -I<头文件目录>：指明头文件的搜索路径
- -iprefix prefix：指定在头文件目录的先前搜索的路径
- -iquote dir：指定用""而不是<>引用的头文件的优先搜索路径
- -isystem*dir*：指定系统头文件的有限搜索路径
- -nostdinc++：不搜索C++的标准头文件路径，搜索其他标准目录
- -X：从搜索路径中去除标准库的路径

重要编译选项 III

- 优化选项
  - -fast：最大化整个程序的速度，相当于：-ipo, -O3, -no-prec-div, -static, 和-xHost。这里是所谓的最大化，还是需要结合程序本身使用合适的选项
  - -inline-level=[n]：设置inline层数
  - -ip：在单个文件中进行过程间优化(Interprocedural Optimizations-IPO)
  - -ipo[n]：在多文件中进行过程间优化，n为可产生的目标文件数，为非负整数
  - -mkl[=lib]：调用MKL中的特定库，lib可以为parallel（线程库，只是-mkl而没有=明确指定时，默认为此）、sequential（线性库）和cluster（集群特征库）
  - -openmp和-qopenmp[1]：编译OpenMP程序，注意：只能在同一个节点的CPU上跑OpenMP程序

重要编译选项 IV

- -O<级别>：设定优化级别，默认为O2，O与O2相同，推荐使用。O3为在O2基础之上增加更激进的优化，比如包含循环和内存读取转换和预取等，但在有些情况下速度反而慢，建议在具有大量浮点计算和大数据处理的循环时的程序使用
- -Od：禁止所有优化
- -Ofast：设置某些激进参数优化程序速度，相当于：-O3 -no-prec-div
- -p：进行概要导向优化(Profile Guided Optimization-PGO)
- -simd和-no-simd：指定是否启用SIMD向量化
- -vec和-no-vec：指定是否启用向量化
- -unroll[n]：循环最大可展开的层数，与性能相关

- 代码生成选项
  - -ax*code*：生成针对Intel处理器的多重特征指定自动派发代码路径。code可以为MIC-AVX512、CORE-AVX512、CORE-AVX2、CORE-AVX-I、AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2等
  - -m*code*：指定为特征目标生成。code可以为MIC-AVX512、CORE-AVX512、CORE-AVX2、CORE-AVX-I、AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2等

重要编译选项 V

- -m32和-m64：指定生成为IA32或Intel 64架构的代码
- -march=<CPU架构>：指定针对某种CPU架构的处理器进行优化。默认为pentium4，可以为generic、core-avx2、core-avx-i、corei7-avx、corei7、atom、core2、pentium*等
- -mtune=<CPU架构>：指定针对某种CPU架构的处理器进行优化。默认为generic，可以为generic、core-avx2、core-avx-i、corei7-avx、corei7、atom、core2、pentium*等
- -x*code*：指定针目标是何种处理特征。默认为generic，可以为CORE-AVX2、CORE-AVX-I、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2等

- 链接选项
  - -Bdynamic：在运行时动态链接所需要的库（.so文件）
  - -Bstatic：静态链接用户生成的库（.a文件）
  - -cxxlib：指定是否链接gcc提供的C++运行时库和头文件，-cxxlib[=dir]、-cxxlib-nostd和-no-cxxlib
  - -L<库目录>：指明库的搜索路径
  - -l<库文件>：指明需链接的库名，如库名为libxyz.a，则可用-lxyz指定

重要编译选项 VI

- -pthread：使用POSIX线程库支持多线程库
- -shared：产生共享目标而不是可执行文件，必须在编译每个目标文件时使用-fpic选项
- -shared-intel：动态链接Intel库
- -shared-libgcc：动态链接GNU libgcc库，可允许用户越过添加静态链接选项-static时的静态链接限制
- -static：静态链接所有库
- -static-intel：静态链接Intel库
- -static-libgcc：动态链接GNU libgcc库，可允许用户越过添加动态链接选项-shared时的动态链接限制
- -static-libstdc++：动态链接GNU libstdc++标准库，可允许用户越过添加动态链接选项-shared时的动态链接限制
- -Wl,optlist：传递以,分割的链接参数给链接器
- -Xlinker val：传递以val变量（如链接参数、目标或库）直接给链接器

重要编译选项 VII

- 语言选项
  - -allow keyword：指明编译器是否运行某些行为，keyword可以为：[no]fpp_comments
  - -altparam和-noaltparam：指明编译器是否允许在PARAMETER声明中不使用括号的代替语法
  - -assume keyword[, keyword...]：指明采用某些默认。keyword可以为：[no]underscore、[no]2underscores等20多种
  - -check [keyword[, keyword...]]和-nocheck：是否对某些条件进行检查，keyword可以为：none、[no]arg_temp_created、[no]assume、[no]bounds、[no]format、[no]output_conversion、[no]pointers、[no]stack、[no]uninit、all
  - -extend-source[size]：指明固定格式的Fortran源代码宽度，size可为72、80和132。也可直接用-72、-80和-132指定，默认为72字符
  - -implicitnone：指明默认变量名为未定义，建议在写程序时添加implicit none语句，以避免出现由于默认类型造成的错误
  - -fixed：指明Fortran源代码为固定格式，默认由文件后缀决定类别
  - -free：指明Fortran源程序为自由格式，默认由文件后缀决定类别

重要编译选项 VIII

- -names keyword：指明源码中标识符和外部名如何翻译，keyword可以为lowercase、uppercase和as_is
- -nofree：指明Fortran源程序为固定格式
- -pad-source和-nopad-source：指明对固定格式源代码是否在行后部补齐空白
- -stand <标准>和-std<标准>：以指定的Fortran标准进行编译，编译时显示源文件中不符合此标准的信息。标准可为f08、f03、f95、f90和none，分别对应显示不符合Fortran 2008、2003、95、90的代码信息和不显示任何非标准的代码信息，也可写为-std<标准>，此时标准不带f，可为08、03、90、95
- -us：编译时给外部用户定义的函数名添加一个下划线，等价于-assume underscore，如果编译时显示_函数找不到时也许添加此选项即可解决
- -stdlib[=keyword]：指定链接所用的C++库。可以为libstdc++（GNU libstdc++库）和libc++（libc++库）
- -strict-ansi：实现严格的ANSI兼容性

重要编译选项 IX

- -x <类型>：类型可以为c、c++、c-header、cpp-output、c++-cpp-output、assembler、assembler-with-cpp或none，分别表示c源文件等，以使所有源文件都被认为是此类型的

- 数据选项
  - -align和-noalign：数据是否自然对齐
  - -auto和-noauto：指明是否所有本地、non-SAVEd的变量分配到运行时堆栈。默认为-auto-scalar
  - -auto-scalar：指明用INTEGER、REAL、COMPLEX、LOGICAL声明时不具有SAVE属性的变量分配到运行时堆栈
  - -save：指明变量存储在静态内存中
  - -convert [关键字]：转换无格式数据的类型，比如关键字为big_endian和little_endian时，分别表示无格式的输入输出为big_endian和little_endian格式，更多格式类型请看编译器手册
  - -check-pointers=keyword：是否检查使用指针访问内存时的数组边界。keyword可以为none、rw和write
  - -dyncom ”common1,common2,...”：指明在运行时动态分配common块
  - -fcommon和-fno-common：指明common块是否为全局定义

重要编译选项 X

- -fpic、-fPIC和-fno-pic：是否生成位置无关代码。当生成共享代码时，必须添加-fpic
- -fpie、-fPIE：与-fpic类似，生成位置无关代码。不同之处在于-fpie生成的代码只能被链入执行程序
- -intconstant和-nointconstant：指明是否用FORTRAN 77语义决定整数的kind参数
- -double-size size：指明对DOUBLE PRECISION和DOUBLE COMPLEX数据类型的默认KIND。可能值为64(KIND=8)或128(KIND=16)
- -integer-size size：指明整型的默认KIND。size可为16、32或64
- -real-size size：指明实型的默认KIND。size可为32、64或128
- -mcmodel=*mem_model*：设定内存模型。mem_model可为：
  - small：限制代码和数据在开始的2GB地址空间，默认
  - medium：限制代码在开始的2GB空间，存储数据空间不受此限制
  - large：对于代码和数据存储空间都无限制
- -zero和-nozero：指明INTEGER、REAL、COMPLEX或LOGICAL声明的本地变量是否初始为0

重要编译选项 XI

- 其它选项
  - –help：显示帮助信息
  - -v：显示详细编译过程以及编译参数等
  - -V：显示编译器版本号
  - -w：编译时不显示任何警告，只显示错误
  - -wall：编译时显示所有警告

[1] 从15系列开始用-qopenmp代替-openmp，-openmp将来会被废弃

串行及OpenMP并行程序编译举例

- 将Fortran 77程序yourprog.for编译为可执行文件yourprog:
  ifort -o yourprog yourprog.for
- 将Fortran 77程序yourprog.for编译为目标文件yourprog.o:
  ifort -c yourprog.for
- 将使用lapack库的Fortran 90程序yourprog.f90编译为可执行文件yourprog:
  ifort -o yourprog -L/opt/lib -llapack yourprog.f90
- 将Fortran 90程序yourprog.f90编译为目标文件yourprog.o:
  ifort -c yourprog.f90
- 将Fortran 90程序yourprog.f90静态编译为O3优化的可执行文件yourprog:
  ifort -O3 -static -o yourprog yourprog.f90
- 将Fortran 90程序yourprog.f90静态编译为可执行文件yourprog:
  ifort -o yourprog -static yourprog.f90
- 将OpenMP指令并行的Fortran 90程序yourprog-omp.f90编译为可执行文件yourprog-omp:
  ifort -o yourprog-omp -openmp yourprog.f90

编译时出错信息格式

```
NOlihm.f90(146): error #6404: This name does not have a type, and must have an explicit type.
[NPR]
  n2nd=0;   npr=0
-----------^
NOlihm.f90(542): remark #8290: Recommended relationship between field width 'W' and the number of
6060 format(/i2,'-th layer',i2,'-th element: z=',i3,' a=',f9.5/' Ef=',f7.5,' (keV)'/' c =',f5.3,'
(at%)')
-----------------------------------------------------------------------^
```

- 源文件名(行数): 错误类型:具体说明
- 源代码，^指示出错位置

运行时错误

根据运行时错误代码在官方手册中查找对应错误解释：

- V15：
  - 逐步链接：
    User and Reference Guide for the Intel®Fortran Compiler 15.0->
    Compiler Reference->Error Handling->
    Handling Run-Time Errors->List of Run-Time Error Messages
  - 直接链接：
    http://scc.ustc.edu.cn/zlsc/tc4600/intel/compiler_f/
    GUID-44448B78-2B87-4998-9828-C8BAEB9F5C9A.htm
- V13：
  - 逐步链接：
    Intel®Fortran Compiler XE 13.0 User and Reference Guides->
    Compiler Reference->Error Handling->
    Handling Run-Time Errors->List of Run-Time Error Messages
  - 直接链接：
    http://scc.ustc.edu.cn/zlsc/chinagrid/intel/compiler_f/main_for/
    GUID-44448B78-2B87-4998-9828-C8BAEB9F5C9A.htm

培训内容

串行程序调试：idb、gdb-ia和gbb-mic

- 利用Intel编译器自带调试器且可以调试OpenMP和MPI并行程序，进入后运行help可以查看具体命令
  - 15版本之前的为Intel自己的调试器：命令行的*idbc*和图形界面的*idb*
  - 从15版本开始采用Intel增强的GNU的调试器：
    - 用于IA-32和Intel 64：*gdb-ia*
    - 用于Intel MIC：*gdb-mic*
- 编译时需要添加*-g*参数，比如：

  ifort -g -o yourprog yourprog.f90
- 启动调试：
  - 无初始调试命令文件方式：*idbc ./yourprog*
  - 有初始调试命令文件方式：*idbc -command dbg.txt ./yourprog*

dbg.txt文件存储调试命令，每行一条命令，启动时自动按顺序执行，其内容类似：

```
break␣139
run
```

详细用法，参见：

- ChinaGrid高性能计算集群使用指南
- Intel®Debugger User's and Reference Guide

联系信息

- 中国科大超算中心:
  - 电话：0551-63602248
  - 信箱：sccadmin@ustc.edu.cn
  - 主页：http://scc.ustc.edu.cn
  - 办公室：中国科大东区新图书馆一楼东侧126室
- 李会民:
  - 电话：0551-63600316
  - 信箱：hmli@ustc.edu.cn
  - 主页：http://hmli.ustc.edu.cn
  - 办公室：中国科大东区新科研楼A座二楼204室