|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | |  |
| **WRF-4.2** | | |  |
| 应用**移植指南** | | |
| **文档版本** | **01** | |
| **发布日期** | **2022-07-30** | |
| 华为网格系统---方案4-032.png | | | | |
|  | 华为技术有限公司 | | 附件1-16K |  |

|  |
| --- |
| 版权所有 © 华为技术有限公司2022。保留一切权利。  非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。  商标声明  附件3-版权声明页图和其他华为商标均为华为技术有限公司的商标。  本文档提及的其他所有商标或注册商标，由各自的所有人拥有。  注意  您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。  由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。 |

|  |  |
| --- | --- |
| 华为技术有限公司 | |
| 地址： | 深圳市龙岗区坂田华为总部办公楼邮编：518129 |
| 网址： | <https://www.huawei.com> |
| 客户服务邮箱： | [support@huawei.com](mailto:support@huawei.com) |
| 客户服务电话： | 4008302118 |

目录

[目录 ii](#_Toc96613985)

[1 介绍 1](#_Toc96613986)

[2 环境要求 2](#_Toc96613987)

[3 移植规划数据 4](#_Toc96613988)

[4 配置编译环境 5](#_Toc96613989)

[4.1搭建鲲鹏基座软件环境 6](#_Toc96613990)

[4.2安装Zlib 6](#_Toc96613991)

[4.3安装HDF-5 6](#_Toc96613992)

[4.4安装PNETCDF 7](#_Toc96613993)

[4.5安装NetCDF 8](#_Toc96613994)

[4.6配置环境变量 9](#_Toc96613995)

[5 获取源码 11](#_Toc96613996)

[6 编译和安装 12](#_Toc96613997)

[7 运行和验证 15](#_Toc96613998)

[8 修订记录 17](#_Toc96613999)

# 介绍

WRF全称 Weather Research and Forecasting Model，是一个天气研究与预报模型，可以用来进行精细尺度的天气模拟与预报，是 HPC 应用的重要场景之一。

关于WRF的更多信息请访问[WRF官网](https://www.mmm.ucar.edu/weather-research-and-forecasting-model)。

语言：C/Fortran

一句话描述：中尺度天气预报模型。

开源协议：Public Domain

建议的版本

建议使用版本为“WRF V4.2”。

# 环境要求

硬件要求

硬件要求如表2-1所示。

硬件要求

| 项目 | 说明 |
| --- | --- |
| CPU | Kunpeng 920 |

软件要求

软件要求如表2-2所示。

注意

* 不同HPC应用的依赖软件不同，建议按照如下步骤判断其依赖软件：
* 1、查看其上游社区是否提供安装指导文档；
* 2、搜索网络上是否已经有社区提供相关安装文档或博客；
* 3、尝试安装该软件，根据报错情况，决定安装哪些依赖软件；
* 4、咨询华为工程师是否有相关经验。

软件要求

| 项目 | 版本 | 下载地址 |
| --- | --- | --- |
| WRF | v4.2 | <https://github.com/wrf-model/WRF/archive/refs/tags/v4.2.tar.gz> |
| 毕昇编译器 | 2.1.0 | https://www.hikunpeng.com/zh/developer/devkit/compiler/bisheng |
| Hyper-MPI | 1.1.1 | <https://github.com/kunpengcompute/hmpi/archive/refs/tags/v1.1.1-huawei.tar.gz> |
| HUCX | 1.1.1 | <https://github.com/kunpengcompute/hucx/archive/refs/tags/v1.1.1-huawei.tar.gz> |
| XUCG | 1.1.1 | <https://github.com/kunpengcompute/xucg/archive/refs/tags/v1.1.1-huawei.tar.gz> |
| Zlib | 1.2.11 | <http://zlib.net/zlib-1.2.11.tar.gz> |
| HDF-5 | 1.12.1 | <https://support.hdfgroup.org/ftp/HDF5/releases/hdf5-1.12/hdf5-1.12.1/src/hdf5-1.12.1.tar.gz> |
| PNETCDF | 1.12.2 | <https://parallel-netcdf.github.io/Release/pnetcdf-1.12.2.tar.gz> |
| NetCDF-C | 4.8.1 | <https://downloads.unidata.ucar.edu/netcdf-c/4.8.1/src/netcdf-c-4.8.1.tar.gz> |
| NetCDF-Fortan | 4.5.3 | <https://www.unidata.ucar.edu/downloads/netcdf/ftp/netcdf-fortran-4.5.3.tar.gz> |
| 测试用例 | conus12km | <https://www2.mmm.ucar.edu/wrf/src/conus12km.tar.gz> |

操作系统要求

操作系统要求如表2-3所示。

操作系统要求

| 项目 | 版本 | 下载地址 |
| --- | --- | --- |
| openEuler | openEuler 20.03 SP3 | https://repo.openeuler.org/openEuler-20.03-LTS-SP3/ |
| Kernel | 4.19.90 | https://gitee.com/openeuler/kernel |

# 移植规划数据

本章节给出WRF软件在移植过程中涉及到的相关软件安装规划路径的用途及详细说明。

移植规划数据

| 序号 | 软件安装规划路径 | 用途 | 说明 |
| --- | --- | --- | --- |
| 1 | /usr/local/wrf/bisheng | 毕昇编译器的安装规划路径。 | 这里的安装规划路径只是一个举例说明，建议部署在共享路径中。需要根据实际情况调整，**后续章节凡是遇到安装路径的命令，都以现网实际规划的安装路径为准进行替换，不再单独说明。** |
| 2 | /usr/local/wrf/hmpi | Hyper-MPI的安装规划路径。 |
| 3 | /usr/local/wrf/zlib | Zlib的安装规划路径。 |
| 4 | /usr/local/wrf/hdf5 | HDF-5的安装规划路径。 |
| 5 | /usr/local/wrf/pnetcdf | PNETCDF的安装规划路径 |
| 6 | /usr/local/wrf/netcdf | NetCDF-C和NetCDF-Fortran的安装规划路径。 |
| 7 | /usr/local/wrf/wrf | WRF的安装规划路径。 |
| 8 | /usr/local/wrf/conus12km | WRF的测试规划路径。 |

# 配置编译环境

前提条件

使用SFTP工具将各安装包上传至服务器对应目录下。

配置流程

配置流程

| 序号 | 配置项 | 说明 |
| --- | --- | --- |
| 1 | 搭建鲲鹏基座软件环境 | 参考4.1 搭建鲲鹏基座软件环境 |
| 2 | 安装Zlib | 参考4.2安装Zlib。 |
| 3 | 安装HDF-5 | 参考4.3安装HDF-5。 |
| 4 | 安装PNETCDF | 参考4.4安装PNETCDF。 |
| 5 | 安装NetCDF | 参考4.5安装NetCDF。 |

## 搭建鲲鹏基座软件环境

操作步骤

请参考《鲲鹏基座软件搭建指南.docx》

## 安装Zlib

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令解压zlib安装包。

**tar -zxvf zlib-1.2.11.tar.gz**

执行以下命令进入解压后的目录。

**cd zlib-1.2.11**

执行以下命令进行配置。

**CC=clang./configure --prefix=/usr/local/wrf/zlib-1.2.11**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

**make -j$(nproc)**

**make -j$(nproc) install**

配置环境变量。

**export ZLIB=/usr/local/wrf/zlib-1.2.11**

**export LD\_LIBRARY\_PATH=$ZLIB/lib:$LD\_LIBRARY\_PATH**

**----结束**

## 安装HDF-5

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令解压HDF-5安装包。

**tar -zxvf hdf5-1.12.1.tar.gz**

执行以下命令进入解压后的目录。

**cd hdf5-1.12.1**

编译HDF-5。

**CC=mpicc CXX=mpicxx FC=mpifort F77=mpi77 CFLAGS="-fPIC -DPIC" FCFLAGS="-fPIC -DPIC" FFLAGS="-fPIC" \**

**./configure --prefix=/usr/local/wrf/hdf5-1.12.1 --with-zlib=/usr/local/wrf/zlib-1.2.11 --enable-shared --enable-fortran --enable-static --enable-parallel**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

**sed -i 's/\\\$wl-soname/-install\_name/g' libtool**

**make-j$(nproc)**

**make -j$(nproc) install**

执行以下命令加载环境变量。

**export HDF5=/usr/local/wrf/hdf5-1.12.1**

**export PATH=$HDF5/bin:$PATH**

**export LD\_LIBRARY\_PATH=$HDF5/lib:$LD\_LIBRARY\_PATH**

----结束

## 安装PNETCDF

操作步骤

使用PuTTY工具，以root用户登陆服务器。

执行以下命令解压PNETCDF 安装包。

**tar xf pnetcdf-1.12.2.tar.gz**

执行以下命令进入解压后的目录。

**cd pnetcdf-1.12.2**

执行以下命令进行配置。

**./configure --prefix=/usr/local/wrf/pnetcdf-1.12.2 --enable-shared CFLAGS="-fPIC -DPIC" CXXFLAGS="-fPIC -DPIC" FCFLAGS="-fPIC" FFLAGS="-fPIC" CC=mpicc CXX=mpicxx FC=mpifort F77=mpifort SEQ\_CC=mpicc**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

**make -j$(nproc)**

**make -j$(nproc) install**

执行以下命令加载环境变量

**export PNETCDF=/usr/local/wrf/pnetcdf-1.12.2**

**export PATH=$PNETCDF/bin:$PATH**

**export LD\_LIBRARY\_PATH=$PNETCDF/lib:$LD\_LIBRARY\_PATH**

---结束

## 安装NetCDF

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令解压NetCDF-C安装包。

**tar -zxvf netcdf-c-4.8.1.tar.gz**

执行以下命令进入解压后的目录。

**cd netcdf-c-4.8.1**

编译NetCDF-C。

**CC=mpicc CXX=mpicxx\**

**CPPFLAGS="-I$HDF5/include -I$ZLIB/include -I$PNETCDF/include" \**

**CFLAGS="-I$HDF5/include -I$ZLIB/include -I$PNETCDF/include-fPIC" \**

**LDFLAGS="-L$HDF5/lib -L$ZLIB/lib -L$PNETCDF/lib" \**

**./configure --prefix=/usr/local/wrf/netcdf --enable-shared --enable-netcdf-4 --enable-dap --with-pic --disable-doxygen--enable-pnetcdf --enable-static --enable-largefile**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

**make-j$(nproc)**

**make -j$(nproc) install**

执行以下命令解压NetCDF-Fortran安装包。

**tar -zxvf netcdf-fortran-4.5.3.tar.gz**

执行以下命令进入解压后的目录。

**cd netcdf-fortran-4.5.3**

编译NetCDF-Fortran。

**export NETCDF=/usr/local/wrf/netcdf**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

**export PATH=$NETCDF/bin:$PATH**

**export LD\_LIBRARY\_PATH="$NETCDF/lib:$LD\_LIBRARY\_PATH"**

**CC=mpicc CXX=mpixxFC=mpifort F77=mpifort\**

**CFLAGS="-I$HDF5/include-I$NETCDF/include -fPIC -DPIC"CPPFLAGS="-I$HDF5/include-I$NETCDF/include" FCFLAGS="-I$HDF5/include -I$NETCDF /include -fPIC -DPIC"LDFLAGS="-L$HDF5/lib -L$NETCDF/lib"**

**./configure --prefix=$NETCDF --enable-dap --enable-pic --disable-doxygen --enable-shared --enable-static --enable-largefile**

**sed -i \**

**-e 's/\\\$wl-soname/-install\_name/g' \**

**-e 's/\\\$wl--whole-archive/-Wl,-force-load,/g' \**

**-e 's/\\\$wl--no-whole-archive//g' libtool**

**make-j$(nproc)**

**make -j$(nproc) install**

执行以下命令加载环境变量。

**export NETCDF=/usr/local/wrf/netcdf**

**export PATH=$NETCDF/bin:$PATH**

**export LD\_LIBRARY\_PATH=$NETCDF/lib:$LD\_LIBRARY\_PATH**

## 配置环境变量

**安装environment-modules工具**

**yum install environment-modules –y**

**source /etc/profile**

**执行以下命令，创建环境变量配置文件**

**vi /usr/local/wrf/wrf\_modulefiles**

**新增如下内容：**

**#%Module1.0**

**conflict wrf**

**set ZLIB/usr/local/wrf/zlib-1.2.11**

**set HDF5/usr/local/wrf/hdf5-1.12.1**

**set PNETCDF/usr/local/wrf/pnetcdf-1.12.2**

**set NETCDF/usr/local/wrf/netcdf**

**setenvZLIB/usr/local/wrf/zlib-1.2.11**

**setenvHDF5/usr/local/wrf/hdf5-1.12.1**

**setenvPNETCDF/usr/local/wrf/pnetcdf-1.12.2**

**setenvNETCDF/usr/local/wrf/netcdf**

**prepend-path PATH$HDF5/bin:$PNETCDF/bin:$NETCDF/bin**

**prepend-path LD\_LIBRARY\_PATH$ZLIB/lib:$HDF5/lib:$PNETCDF/lib:$NETCDF/lib**

**执行以下命令，在当前shell中加载环境变量**

**module use /usr/local/wrf/**

**module load /usr/local/wrf/wrf\_modulefiles**

**若要避免每打开一个shell，导入一次变量。可写入到系统配置文件中。**

**vi /etc/profile**

**新增如下内容：**

**module use /usr/local/wrf/**

**module load /usr/local/wrf/wrf\_modulefiles**

----结束

# 获取源码

操作步骤

下载WRF安装包“v4.2.tar.gz”。

下载地址：<https://github.com/wrf-model/WRF/archive/refs/tags/v4.2.tar.gz>。

----结束

# 编译和安装

操作步骤

使用PuTTY工具，以root用户登录服务器。

执行以下命令解压WRF安装包。

**tar -zxvfv4.2.tar.gz –C /usr/local/wrf/**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

执行以下命令进入WRF源码所在目录。

**cd /usr/local/wrf/&& mv WRF-4.2wrf-4.2**

**cd wrf-4.2/**

制作Patch补丁。

**vi configure.defaults.patch**

写入以下内容并保存：

--- ./arch/configure.defaults 2020-04-24 01:08:37.000000000 +0800

+++ ../WRF-4.2.2/arch/configure.defaults 2021-01-16 01:21:58.000000000 +0800

@@ -44,7 +44,7 @@

CC\_TOOLS = cc

###########################################################

-#ARCH Linux i486 i586 i686, gfortran compiler with gcc #serial smpardmpardm+sm

+#ARCH Linux i486 i586 i686 armv7l aarch64, gfortran compiler with gcc #serial smpardmpardm+sm

#

DESCRIPTION = GNU ($SFC/$SCC)

DMPARALLEL = # 1

@@ -73,8 +73,9 @@

FORMAT\_FIXED = -ffixed-form

FORMAT\_FREE = -ffree-form -ffree-line-length-none

FCSUFFIX =

+FCCOMPAT =

BYTESWAPIO = -fconvert=big-endian -frecord-marker=4

-FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO)

+FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO) $(FCCOMPAT)

FCBASEOPTS = $(FCBASEOPTS\_NO\_G) $(FCDEBUG)

MODULE\_SRCH\_FLAG =

TRADFLAG = CONFIGURE\_TRADFLAG

@@ -785,8 +786,9 @@

FORMAT\_FIXED = -ffixed-form

FORMAT\_FREE = -ffree-form -ffree-line-length-none

FCSUFFIX =

+FCCOMPAT =

BYTESWAPIO = -fconvert=big-endian -frecord-marker=4

-FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO)

+FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO) $(FCCOMPAT)

FCBASEOPTS = $(FCBASEOPTS\_NO\_G) $(FCDEBUG)

MODULE\_SRCH\_FLAG =

TRADFLAG = CONFIGURE\_TRADFLAG

@@ -1006,8 +1008,9 @@

FORMAT\_FIXED = -ffixed-form

FORMAT\_FREE = -ffree-form -ffree-line-length-none

FCSUFFIX =

+FCCOMPAT =

BYTESWAPIO = -fconvert=big-endian -frecord-marker=4

-FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO)

+FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO) $(FCCOMPAT)

FCBASEOPTS = $(FCBASEOPTS\_NO\_G) $(FCDEBUG)

MODULE\_SRCH\_FLAG =

TRADFLAG = CONFIGURE\_TRADFLAG

@@ -1049,8 +1052,9 @@

FORMAT\_FIXED = -ffixed-form

FORMAT\_FREE = -ffree-form -ffree-line-length-none

FCSUFFIX =

+FCCOMPAT =

BYTESWAPIO = -fconvert=big-endian -frecord-marker=4

-FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO)

+FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO) $(FCCOMPAT)

FCBASEOPTS = $(FCBASEOPTS\_NO\_G) $(FCDEBUG)

MODULE\_SRCH\_FLAG =

TRADFLAG = CONFIGURE\_TRADFLAG

@@ -1772,8 +1776,9 @@

FORMAT\_FIXED = -ffixed-form

FORMAT\_FREE = -ffree-form -ffree-line-length-none

FCSUFFIX =

+FCCOMPAT =

BYTESWAPIO = -fconvert=big-endian -frecord-marker=4

-FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO)

+FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO) $(FCCOMPAT)

FCBASEOPTS = $(FCBASEOPTS\_NO\_G) $(FCDEBUG)

MODULE\_SRCH\_FLAG =

TRADFLAG = CONFIGURE\_TRADFLAG

@@ -1829,7 +1834,7 @@

CC\_TOOLS = $(SCC)

###########################################################

-#ARCH Linux x86\_64 ppc64le i486 i586 i686 #serial smpardmpardm+sm

+#ARCH Linux x86\_64 ppc64le i486 i586 i686, ifort compiler with icc #serial smpardmpardm+sm

#

DESCRIPTION = INTEL ($SFC/$SCC): HSW/BDW

DMPARALLEL = # 1

@@ -1873,7 +1878,7 @@

CC\_TOOLS = $(SCC)

###########################################################

-#ARCH Linux KNL x86\_64 ppc64le i486 i586 i686 #serial smpardmpardm+sm

+#ARCH Linux KNL x86\_64 ppc64le i486 i586 i686, ifort compiler with icc #serial smpardmpardm+sm

#

DESCRIPTION = INTEL ($SFC/$SCC): KNL MIC

DMPARALLEL = # 1

@@ -1946,8 +1951,9 @@

FORMAT\_FIXED = -ffixed-form

FORMAT\_FREE = -ffree-form -ffree-line-length-none

FCSUFFIX =

+FCCOMPAT =

BYTESWAPIO = -fconvert=big-endian -frecord-marker=4

-FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO)

+FCBASEOPTS\_NO\_G = -w $(FORMAT\_FREE) $(BYTESWAPIO) $(FCCOMPAT)

FCBASEOPTS = $(FCBASEOPTS\_NO\_G) $(FCDEBUG)

MODULE\_SRCH\_FLAG =

TRADFLAG = -traditional

为WRF打上补丁：

**yum install csh patch time -y**

**patch -p1 < ./configure.defaults.patch**

**echo 4 | ./configure**

**sed -i 's/derf/erf/g' ./phys/module\_mp\_SBM\_polar\_radar.F**

**sed -i -e 's/gcc/clang/g' \**

**-e 's/gfortran/flang/g' \**

**-e 's/mpicc/mpicc -DMPI2\_SUPPORT/g' \**

**-e 's/ -ftree-vectorize//g' \**

**-e 's/length-none/length-0/g' \**

**-e 's/-frecord-marker\=4/ /g' \**

**-e 's/\-w \-O3 \-c/-mcpu=native \-w \-O3 \-c/g' \**

**-e 's/\# \-g $(FCNOOPT).\*/\-g/g' \**

**-e 's/$(FCBASEOPTS\_NO\_G)/-mcpu=native $(OMP) $(FCBASEOPTS\_NO\_G)/g' configure.wrf**

**ln -sf /usr/lib64/libz.so.1.2.11 /usr/lib64/libz.so**

执行以下命令进行编译。

**./compile -j $(nproc)em\_real | tee compile.log**

----结束

# 运行和验证

操作步骤

使用PuTTY工具，以root用户登录服务器。

下载WRF测试用例“conus12km”。

下载地址：<https://www2.mmm.ucar.edu/wrf/src/conus12km.tar.gz>。

使用SFTP工具将测试用例上传至服务器*“/path/to/conus12km”*目录。

解压测试数据。

**tar xf conus12km.tar.gz --strip 1 -C /usr/local/wrf/conus12km**

注：*/usr/local/wrf/为举例说明路径，请根据用例存放实际路径修改。*

执行以下命令进入工作目录并把测试用例和二进制文件拷贝到工作目录。

**cp -r /usr/local/wrf/wrf-4.2/run/\*c**onus12km

提示覆盖namelist.input时，请选择“否”。

**cd conus12km**

**ln -sf/usr/local/wrf/wrf-4.2/main/\*.exe .**

若进行单节点测试，则执行以下命令：

**mpirun -np64 ./wrf.exe**

注：*-np64*指定使用的CPU核数，请根据实际情况修改。

若使用root用户操作时报错，请尝试执行：

**mpirun --allow-run-as-root -np** 64 **./wrf.exe**

若进行多节点测试

多节点测试以双节点为例，前提说明：

1、双节点算例运行在共享目录中，如第四章节配置编译中已配置，无需再配置PATH，LD\_LIBRARY\_PATH环境变量

2、相关参数

|  |  |
| --- | --- |
| **参数** | **说明** |
| **-np** | 指mpi运行的总进程数 |
| **-N** | 每个节点上运行的进程数 |
| **--hostfile** | 使用的节点名字 |

3、hostfile文件中放入自己指定的两个计算节点(如*Node1*和*Node2*,其中*Node1*和*Node2*均为主机名称，保证节点相互可以ssh切换成功)，如下所示:

Node1

Node2

执行以下命令运行WRF程序：

**mpirun--allow-run-as-root–hostfile**hostfile**-x PATH=$PATH -x LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH-np** *256* ***–N*** *128* **./wrf.exe**

要验证WRF正确运行，请执行：

**tail rsl.out.0000**

若输出中出现：

wrf: SUCCESS COMPLETE WRF

则说明WRF测试成功完成。

----结束

# 修订记录

| 发布日期 | 修订记录 |
| --- | --- |
| 2022-07-30 | 第一次正式发布。 |