|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | |  |
| **鲲鹏基座软件** | | |  |
| **搭建指南** | | |
| **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

[1 介绍 1](#_Toc118469078)

[2 环境要求 2](#_Toc118469079)

[3 移植规划数据 4](#_Toc118469080)

[4 安装毕昇编译器 5](#_Toc118469081)

[4.1 yum源安装方式 6](#_Toc118469082)

[4.2 软件包安装方式 6](#_Toc118469083)

[5 安装Hyper-MPI 9](#_Toc118469084)

[5.1 毕昇编译hucx软件包 10](#_Toc118469085)

[5.2 编译带CUDA-aware MPI功能的HUCX 10](#_Toc118469086)

[5.3 毕昇编译Hyper-MPI软件包 14](#_Toc118469087)

[6 修订记录 17](#_Toc118469088)

# 介绍

鲲鹏HPC基座软件包括毕昇编译器、Hyper-MPI高性能通信库等，其以鲲鹏服务器为硬件基础，结合芯片特点，构建了各场景解决方案计算底层生态。

以Hyper-MPI为例，目前业界领先的MPI库，均采用闭源发布策略，且不支持鲲鹏生态；Hyper MPI则不仅可以在x86服务器上使能，而且可以在以鲲鹏芯片为核心的服务器和集群上发挥作用。且相比于x86服务器，鲲鹏服务器具有单节点核数多、指令集简单、耗能低、价格实惠等特点，因此这有利于构建以鲲鹏芯片为硬件核心的计算生态。

关于鲲鹏生态的更多信息请访问鲲鹏官网。

# 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。

软件要求

| 项目 | 版本 | 下载地址 |
| --- | --- | --- |
| 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> |

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

本章节给出各基座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 | */workspace/home/migration/zhusihai/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 | */workspace/home/migration/zhusihai/hmpi* | Hyper-MPI的安装规划路径。 |

# 安装毕昇编译器

本节介绍毕昇编译器的安装步骤，以下操作均使用root用户执行。毕昇编译器已经融入openEuler源，在openEuler操作系统中，可以使用yum源方式安装毕昇编译器；在非openEuler操作系统中，可以通过软件包方式安装毕昇编译器。

环境依赖

* 内存：8GB以上
* 操作系统：openEuler21.03、openEuler 20.03 (LTS)、CentOS 7.6、Ubuntu 18.04、Ubuntu 20、麒麟V10、UOS 20
* 架构：AArch64
* GCC版本：4.8.5以上
* glibc版本：2.17以上
* libatomic版本：1.2及以上

## yum源安装方式

**步骤1**在/etc/yum.repos.d/目录下增加配置文件bisheng-compiler.repo，运行如下命令：

**cat> /etc/yum.repos.d/bisheng-compiler.repo<< EOF**

**[bisheng-compiler]**

**name=bisheng-compiler**

**baseurl=https://repo.oepkgs.net/bisheng/aarch64/**

**enabled=1**

**gpgcheck=0**

**priority=100**

**EOF**

**步骤2**从yum源下载和安装毕昇编译器rpm包

**yum update**

**yum install bisheng-compiler**

**步骤3**（可选）清空当前窗口的hash表

如果系统中有其他版本的 LLVM 编译器，请在安装毕昇编译器之后立即运行如下命令

**hash -r**

防止clang命令被hash捕获，出现毕昇编译器或开源LLVM编译器无法使用的问题。

**步骤4**验证安装是否成功

安装完毕后执行如下命令验证毕昇编译器版本：

**clang -v**

若返回结果已包含bisheng compiler版本信息，说明安装成功。

## 软件包安装方式

**步骤1**准备工作

在毕昇编译器产品页选择“毕昇编译器软件包下载”获取毕昇编译器软件包。且已上传至服务器上。

https://www.hikunpeng.com/zh/developer/devkit/compiler/bisheng

**步骤2**设置安装目录

1. 创建毕昇编译器安装目录

**mkdir -p /workspace/home/migration/zhusihai/bisheng**

注：**/workspace/home/migration/zhusihai/***为举例说明路径，请根据用例存放实际路径修改。*

2. 将毕昇编译器压缩包拷贝到安装目录下，并解压：

**cp bisheng-compiler-2.1.0-aarch64-linux.tar.gz /workspace/home/migration/zhusihai/bisheng**

**cd /workspace/home/migration/zhusihai/bisheng**

**tar -zxvf bisheng-compiler-2.1.0-aarch64-linux.tar.gz**

解压完成后在当前目录下出现名为**bisheng-compiler-2.1.0-aarch64-linux**的目录。

**步骤3**配置毕昇编译器的环境变量

* 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 /workspace/home/migration/zhusihai/**bisheng/bisheng\_modulefiles**

**新增如下内容：**

**#%Module1.0**

**conflict bisheng**

**prepend-path PATH /workspace/home/migration/zhusihai/bisheng/bisheng-compiler-2.1.0-aarch64-linux/bin**

**prepend-path LD\_LIBRARY\_PATH /workspace/home/migration/zhusihai/bisheng/bisheng-compiler-2.1.0-aarch64-linux/lib**

* **执行以下命令，在当前shell中加载环境变量**

**module use** /workspace/home/migration/zhusihai/**bisheng/**

**module load** /workspace/home/migration/zhusihai/**bisheng/bisheng\_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** /workspace/home/migration/zhusihai/**bisheng/**

**module load** /workspace/home/migration/zhusihai/**bisheng/bisheng\_modulefiles**

**步骤4**（可选）清空当前窗口的hash表

如果系统中有其他版本的 LLVM 编译器，请在安装毕昇编译器之后立即运行如下命令

**hash -r**

防止clang命令被hash捕获，出现毕昇编译器或开源LLVM编译器无法使用的问题。

**步骤5**验证安装是否成功

安装完毕后执行如下命令验证毕昇编译器版本：

**clang -v**

若返回结果已包含bisheng compiler版本信息，说明安装成功。

# 安装Hyper-MPI

Hyper MPI包含Hyper MPI和HUCX两个软件层，其中Hyper MPI的安装依赖于HUCX，编译时应先编译HUCX。

本安装将用HMPI库的多线程（--enable-mpi-thread-multiple, OpenMPI 2.0版本才需要启动此参数）和GPUDirect RDMA功能。

前提条件

已完成以下依赖包的安装。

* **perl-Data-Dumper**
* **autoconf**
* **automake**
* **libtool 2.4.2**
* **毕昇编译器2.1.0**
* **numactl**
* **binutils**
* **systemd-devel**
* **valgrind**
* **flex**

以下是安装GPUDirect RDMA的依赖包。

* **GDRCopy 2.3**
* **Mellanox OFED Driver 5.4**
* **nv-peer-memory 1.1-0**

## 毕昇编译hucx软件包

**步骤1** 下载HUCX源码包“hucx-1.1.1-huawei.tar.gz”。

下载地址：<https://github.com/kunpengcompute/hucx/archive/refs/tags/v1.1.1-huawei.tar.gz>

**步骤2** 下载XUCG源码包“xucg-1.1.1-huawei.tar.gz”。

下载地址：<https://github.com/kunpengcompute/xucg/archive/refs/tags/v1.1.1-huawei.tar.gz>

**步骤3** 将上述两个源码包上传至服务器HUCX源码包安装目录，如“/path/to/package/hucx”，执行以下命令，解压上述两个源码包。

**tar -zxvf hucx-1.1.1-huawei.tar.gz**

**tar -zxvf xucg-1.1.1-huawei.tar.gz**

**步骤4**执行以下命令，将XUCG源码包中的内容复制到HUCX源码包中的“src/ucg”目录下。

**cp -r xucg-1.1.1-huawei/\* hucx-1.1.1-huawei/src/ucg**

**步骤5**执行以下命令，进入“autogen.sh”脚本所在目录，并初始化git仓库。

**cd hucx-1.1.1-huawei/ && git init**

**步骤6**执行以下命令，进行源码包安装。

**./autogen.sh**

**./contrib/configure-opt -prefix=/workspace/home/migration/zhusihai/hmpi/hucx CC=clang CXX=clang++ FC=flang**

注：*/usr/local/为举例说明路径，请根据用例存放实际路径修改。*

*/usr/local/在登录节点上非超级用户没有权限，更改为/workspace/home/migration/zhusihai/*

注意

* 环境上若没有安装numactl插件，则需要执行以下命令，指定--disable-numa参数：
* **./contrib/configure-opt -prefix=/workspace/home/migration/zhusihai/hmpi/hucx --disable-numa CC=clang CXX=clang++ FC=flang**

## 编译带CUDA-aware MPI功能的HUCX

如果需要安装CUDA-aware MPI库，则需要安装下面的软件。

**步骤1 安装GDRCopy**

**wget**

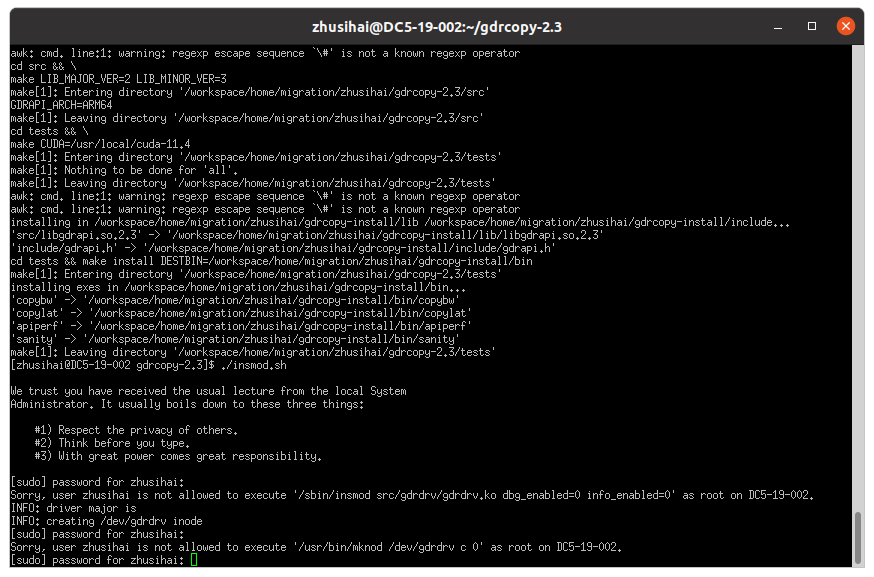
**tar xf**

**export MY\_PATH=/workspace/home/migration/zhusihai**

**make prefix=$MY\_PATH/gdrcopy\_install CUDA=/usr/local/cuda-11.4 all install**

**./insmod.sh**

注：执行./insmod.sh，需要超级用户(su)权限。在鲲鹏登录节点上，没有su权限，出现错误：

****

**步骤2 安装OFED驱动程序**

鲲鹏集群的登录节点上已经安装了Mellanox网卡的硬件和软件驱动，不过目前网卡是处于Ethernet状态。如果需要使用GPUDirect RDMA功能，需要切换至IB模式。

ibv\_devinfo // 查看是否安装OFED驱动，如果没有安装，需要下载驱动并安装。

Infiniband卡支持两种工作模式：IB模式和Ethernet模式。

首先，启动mst工具，通过mst工具查看自己的MST devices：/dev/mst/mt4119\_pciconf0 （没有mst工具，需要下载安装）

[root@localhost ~]# systemctl start mst

[root@localhost ~]# mst status

MST modules:

------------

MST PCI module is not loaded

MST PCI configuration module loaded

MST devices:

------------

/dev/mst/mt4119\_pciconf0 - PCI configuration cycles access.

domain:bus:dev.fn=0000:c4:00.0 addr.reg=88 data.reg=92 cr\_bar.gw\_offset=-1

Chip revision is: 00

[root@localhost ~]#

查看网卡的配置信息，在下列输出中，可以看出此时网卡的工作模式是IB模式，也可以通过ifconfig去查看。

[root@localhost ~]# mlxconfig -d /dev/mst/mt4119\_pciconf0 query

Device #1:

----------

Device type: ConnectX5

Name: MCX555A-ECA\_Ax

Description: ConnectX-5 VPI adapter card; EDR IB (100Gb/s) and 100GbE; single-port QSFP28; PCIe3.0 x16; tall bracket; ROHS R6

Device: /dev/mst/mt4119\_pciconf0

Configurations: Next Boot

MEMIC\_BAR\_SIZE 0

MEMIC\_SIZE\_LIMIT \_256KB(1)

HOST\_CHAINING\_MODE DISABLED(0)

[root@localhost ~]#

修改网卡的工作模式：

Ethernet模式： mlxconfig -d /dev/mst/mt4119\_pciconf0 set LINK\_TYPE\_P1=2

IB模式： mlxconfig -d /dev/mst/mt4119\_pciconf0 set LINK\_TYPE\_P1=1

[root@localhost ~]# mlxconfig -d /dev/mst/mt4119\_pciconf0 set LINK\_TYPE\_P1=2

Device #1:

----------

Device type: ConnectX5

Name: MCX555A-ECA\_Ax

Description: ConnectX-5 VPI adapter card; EDR IB (100Gb/s) and 100GbE; single-port QSFP28; PCIe3.0 x16; tall bracket; ROHS R6

Device: /dev/mst/mt4119\_pciconf0

Configurations: Next Boot New

LINK\_TYPE\_P1 IB(1) ETH(2)

Apply new Configuration? (y/n) [n] : y

Applying... Done!

-I- Please reboot machine to load new configurations.

[root@localhost ~]#

如上，mlxconfig -d /dev/mst/mt4119\_pciconf0 set LINK\_TYPE\_P1=2，网卡已修改为Ethernet模式，修改后需要重启机器后才生效，因为修改的是下一次(Next boot)的网卡启动的工作模式。

**步骤3 安装nv\_peer\_memory**

用于GPU设备间的直接显存拷贝。

wget nv\_peer\_memory-1.1-0.tar.gz

tar xf nv\_peer\_memory-1.1-0.tar.gz //由于鲲鹏登录节点的OFED版本与1.2版本不兼容，因此使用1.1-0版本的nv\_peer\_memory

cd nvidia\_peer\_memory-1.1-0

./build\_module.sh

rpmbuild --rebuild /tmp/nvidia\_peer\_memory-1.1-0.src.rpm

rpm -ivh /workspace/home/migration/zhusihai/rpmbuild/RPMS/aarch64/nvidia\_peer\_memory-1.1-0.ky10.aarch64.rpm

**-------------------------------**

**问题、原因与解决：**

**-------------------------------**

出现错误：

/workspace/home/migration/zhusihai/rpmbuild/BUILD/nvidia\_peer\_memory-1.3/nv\_peer\_mem.c: In function 'nv\_mem\_client\_init':

/workspace/home/migration/zhusihai/rpmbuild/BUILD/nvidia\_peer\_memory-1.3/nv\_peer\_mem.c:544:25: error: invalid use of undefined type 'struct peer\_memory\_client\_ex'

strcpy(nv\_mem\_client\_ex.client.name, DRV\_NAME);

.......

解决方法：

https://github.com/Mellanox/nv\_peer\_memory/issues/94

Mellanox driver: MLNX\_OFED\_LINUX-5.4-1.0.3.0出现上述一样的问题。

MLNX\_OFED\_LINUX-5.4-3.0.3.0驱动解决了这个问题。

或者

使用nv\_peer\_memory-1.1-0.tar.gz

**步骤4 安装带GDRCopy的HUCX库**

**./contrib/configure-opt -prefix=$MY\_PATH/hmpi/hucx --disable-numa --with-cuda=/usr/local/cuda-11.4 --with-gdrcopy=$MY\_PATH/gdrcopy-install CC=clang CXX=clang++ FC=flang**

**步骤5修改编译UCX的Makefile**

**for file in `find . -name Makefile`;do sed -i "s/-Werror//g" $file;done**

**for file in `find . -name Makefile`;do sed -i "s/-implicit-function-declaration//g" $file;done**

**make -j$(nproc)**

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

**-------------------------------**

**问题、原因与解决：**

**-------------------------------**

问题：/usr/bin/ld: cannot find -lgdrapi

clang-12: error: linker command failed with exit code 1 (use -v to see invocation)

原因： 生成的Makefile中：GDR\_COPY\_CPPFLAGS = -I/workspace/home/migration/zhusihai/gdrcopy-install/include/

GDR\_COPY\_LDFLAGS = -lgdrapi -L/workspace/home/migration/zhusihai/gdrcopy-install/lib64

因此，将gdrcopy-install/lib 修改为 gdrcopy-install/lib64

执行命令：mv $MY\_PATH/gdrcopy-install/lib $MY\_PATH/gdrcopy-install/lib64

**步骤7**配置环境变量

**export PATH=/workspace/home/migration/zhusihai/hmpi/hucx/bin:$PATH**

**export LD\_LIBRARY\_PATH=/workspace/home/migration/zhusihai/hmpi/hucx/lib:$LD\_LIBRARY\_PATH**

**export INCLUDE=/workspace/home/migration/zhusihai/hmpi/hucx/include:$INCLUDE**

## 毕昇编译Hyper-MPI软件包

**步骤1**下载Hyper MPI源码包“hmpi-1.1.1-huawei.tar.gz”。

下载地址：https://github.com/kunpengcompute/hmpi/archive/refs/tags/v1.1.1-huawei.tar.gz

**步骤2**将源码包上传至服务器Hyper MPI源码包安装目录，如“/path/to/package/hmpi”；执行以下命令，解压软件包。

**tar -zxvf hmpi-1.1.1-huawei.tar.gz**

**步骤3**执行以下命令，进入“autogen.pl”脚本所在目录。

**cd hmpi-1.1.1-huawei/**

**步骤4**执行以下命令，进行Hyper MPI源码包安装。

**./autogen.pl CC=clang CXX=clang++ FC=flang**

**./configure --prefix=/workspace/home/migration/zhusihai/hmpi --with-platform=contrib/platform/mellanox/optimized --enable-mpi1-compatibility --with-ucx=/workspace/home/migration/zhusihai/hmpi/hucx CC=clang CXX=clang++ FC=flang**

注：**/workspace/home/migration/zhusihai/***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**

**步骤5配置环境变量**

* **执行以下命令，创建环境变量配置文件**

**vi /workspace/home/migration/zhusihai/hmpi/hmpi\_modulefiles**

**新增如下内容：**

**#%Module1.0**

**conflict hmpi**

**set OPAL\_PREFIX /workspace/home/migration/zhusihai/hmpi**

**setenv OPAL\_PREFIX /workspace/home/migration/zhusihai/hmpi**

**prepend-path PATH $OPAL\_PREFIX/bin:$OPAL\_PREFIX/hucx/bin**

**prepend-path LD\_LIBRARY\_PATH $OPAL\_PREFIX/lib:$OPAL\_PREFIX/hucx/lib**

**prepend-path INCLUDE $OPAL\_PREFIX/include:$OPAL\_PREFIX/hucx/include**

* **执行以下命令，在当前shell中加载环境变量**

**module use /workspace/home/migration/zhusihai/hmpi/**

**module load /workspace/home/migration/zhusihai/hmpi/hmpi\_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 /workspace/home/migration/zhusihai/hmpi/**

**module load /workspace/home/migration/zhusihai/hmpi/hmpi\_modulefiles**

**步骤6 安装后检查：**执行以下命令，进行安装后检查。

**which mpirun**

若返回如下安装路径示例，则证明安装成功。

**/workspace/home/migration/zhusihai/hmpi/bin/mpirun**

**步骤7** 运行支持GPUDirect RDMA的应用程序

mpirun --mca btl\_openib\_want\_cuda\_gdr 1 ...

例如mgcfd的多GPU运行脚本：

mpirun -hostfile $HOSTFILE -np 2 --mca plm\_rsh\_agent /opt/batch/agent/tools/dstart -x LD\_LIBRARY\_PATH -mca pml ucx -x UCX\_NET\_DEVICES=mlx5\_0:1 -mca btl ^vader,tcp,openib,uct -x UCX\_TLS=self,sm,rc,cuda\_copy, cuda\_ipc --bind-to core --map-by socket --rank-by core mgcfd\_mpi\_cuda -i input.dat -m parmetis -r kway

UCX\_TLS可用于调整MPI使用的通信协议。UCX\_TLS可以添加：cuda\_copy, gdr\_copy (openMPI的参数，HMPI呢？)，实现多GPU间的GPUDirect RDMA数据传输。

HMPI UCX\_TLS可参考：<https://support.huawei.com/enterprise/zh/doc/EDOC1100216760?section=j03a>

**-------------------------------**

**问题、原因与解决：**

**-------------------------------**

目前在鲲鹏上，正确安装和配置了具有GPUDirect RDMA功能的OpenMPI-4.1通信库。

运行：

ompi\_info --parsable --all | grep mpi\_built\_with\_cuda\_support:value

ompi\_info --all | grep btl\_openib\_have\_cuda\_gdr

ompi\_info --all | grep btl\_openib\_have\_driver\_gdr

均返回true，表明驱动和具备GDR功能的MPI通信库安装成功。但要正确执行GDR环境下的可执行程序运行，还需要做如下配置：

**问题1**：执行带参数UCX\_TLS=self,sm,rc,cuda\_copy, cuda\_ipc的脚本后，出现错误：

UCX WARN transport 'cuda\_copy' is not available, please use one or more of: cma, dc, dc\_mlx5, dc\_x, ib, mm, posix, rc, rc\_mlx5, rc\_v, rc\_verbs, rc\_x, self, shm, sm, sysv, tcp, ud, ud\_mlx5, ud\_v, ud\_verbs, ud\_x

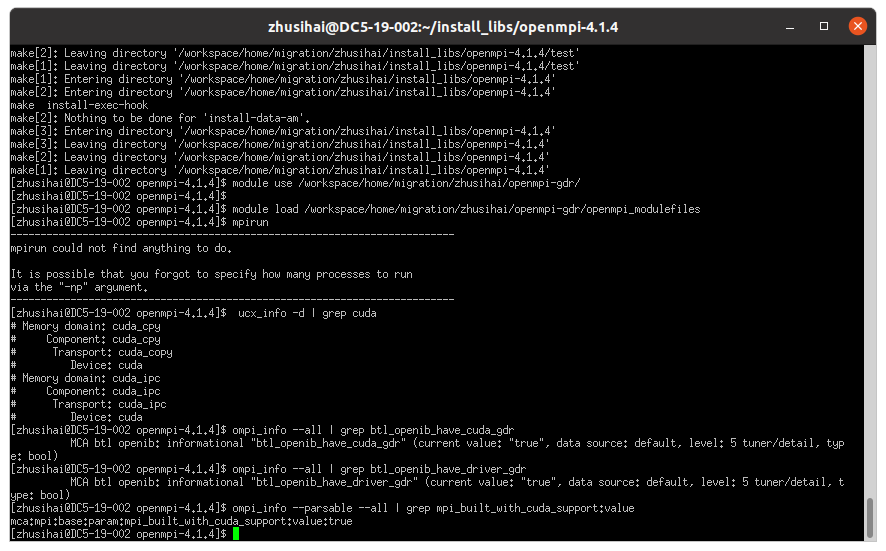
解决方法：

参考： https://github.com/openucx/ucx/issues/3359

https://github.com/openucx/ucx/issues/7917

执行命令：ucx\_info -d | grep cuda

显示：



执行命令：lsmod | grep gdr

显示：

gdrdrv 262144 0

nvidia 35586048 21 nvidia\_uvm,nv\_peer\_mem,gdrdrv,nvidia\_modeset

执行命令：UCX\_LOG\_LEVEL=debug ucx\_info -d | grep gdr

显示：

[1667738416.156190] [DC5-19-002:776838:0] init.c:119 UCX DEBUG /workspace/home/migration/zhusihai/openmpi-gdr/ucx/lib/libucs.so.0 loaded at 0x40002ff10000

[1667738416.156216] [DC5-19-002:776838:0] module.c:72 UCX DEBUG ucs library path: /workspace/home/migration/zhusihai/openmpi-gdr/ucx/lib/libucs.so.0

[1667738416.180981] [DC5-19-002:776838:0] module.c:165 UCX DEBUG ignoring 'ucs\_module\_global\_init' (0x400030356b08) from libuct\_cuda.so.0 (0x400030350000), expected in libuct\_cuda\_gdrcopy.so.0 (400031f80000)

[1667738416.262568] [DC5-19-002:776838:0] gdr\_copy\_md.c:429 UCX DEBUG could not create registration cache for: Unsupported operation

# Memory domain: gdr\_copy

# Component: gdr\_copy

# Transport: gdr\_copy

解决方法：

export LD\_PRELOAD="/workspace/home/migration/zhusihai/gdrcopy-install/lib64/libgdrapi.so"

执行命令：ucx\_info -d | grep Transport

# Transport: posix

# Transport: sysv

# Transport: self

# Transport: tcp

# Transport: tcp

# Transport: tcp

# Transport: tcp

# Transport: cuda\_copy

# Transport: cuda\_ipc

# Transport: rc\_verbs

# Transport: rc\_mlx5

# Transport: dc\_mlx5

# Transport: ud\_verbs

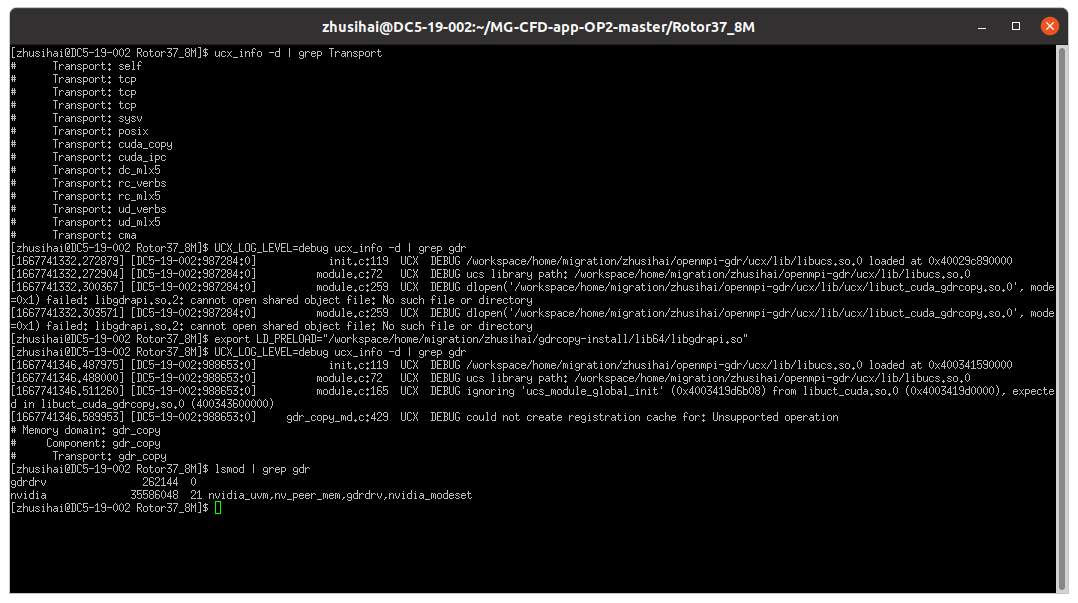
# Transport: ud\_mlx5

# Transport: cma

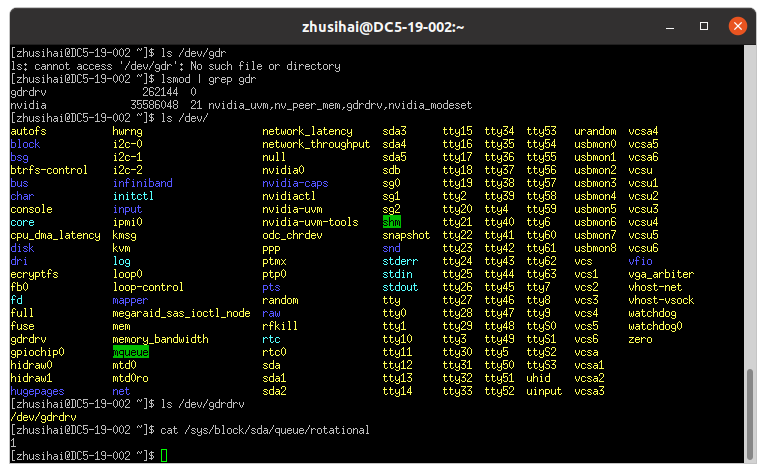
# Transport: knem

# Transport: xpmem

最终，全部设置正确后，运行上述命令，屏幕显示将如下图：



另外，鲲鹏工作站上没有使用NVMe SSD，还是机械硬盘，因此不能使用GDS。



----结束

# 修订记录

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