CANN
V100R020C10

Ascend-DMI 工具用户指南

文档版本 01

发布日期 2021-01-12

华为技术有限公司

商标声明


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。

目 录

1 简介

工具介绍

“Ascend-DMI”工具主要为Atlas产品的标卡、板卡及模组类产品提供带宽测试、算力测试、功耗测试等功能。工具的功能介绍如表1-1所示。本系统通过调用底层DCMI（设备控制管理接口）/DSMI（设备系统管理接口）以及ACL（Ascend Computing Language，昇腾计算语言）相关接口完成相关检测功能，对于系统级别的信息查询通过调用系统提供的通用库来实现，用户使用工具时通过配置参数来实现不同的测试功能。

表 1-1 工具功能介绍

功能名称	功能介绍
带宽测试	测试总线带宽、内存带宽和时延。
算力测试	测试芯片中AI Core的算力值和满算力下芯片的平均功率。
功耗测试	检测整卡或芯片的功耗信息。
设备实时状态查询	检测设备在运行过程中的状态信息。
故障诊断	获取芯片健康信息，同时对芯片进行算力、功耗、带宽测试并输出测试结果。
软硬件版本兼容性测试	获取硬件信息、架构、驱动版本、固件版本以及软件版本，并检测软硬件间的兼容性。
设备拓扑检测	查询单机内多卡间的拓扑结构。

约束与限制

Profiling性能分析工具开启时，“Ascend-DMI”工具不可用。

Profiling性能分析工具请参考如下手册：

推理场景：请参考《CANN V100R020C10 开发辅助工具指南（推理）》的“**使用约束与环境准备**”章节。

训练场景：请参考《CANN V100R020C10 开发辅助工具指南（训练）》的“**概述**”章节。

异构计算架构介绍

在异构计算架构中，昇腾AI处理器与CPU通过PCIe总线连接协同工作，我们一般分别称为Device侧和Host侧，具体说明如下：

- Host：CPU所在位置称为主机端（Host），是指与昇腾AI处理器所在硬件设备（如Atlas 300I 推理卡）相连接的X86或Arm服务器，利用昇腾AI处理器提供的加速计算能力完成业务。
- Device：是指安装了昇腾AI处理器的硬件设备，利用PCIe接口与服务器连接，为服务器提供加速计算能力。

环境适配信息

“Ascend-DMI”工具适配的硬件形态如**表1-2**所示。

表 1-2 适配信息

硬件形态
Atlas 800 推理服务器（型号 3000）
Atlas 800 推理服务器（型号 3010）
Atlas 800 训练服务器（型号 9000）
Atlas 800 训练服务器（型号 9010）
Atlas 300T 训练卡（型号 9000）
Atlas 300I 推理卡（型号 3000）
Atlas 300I 推理卡（型号 3010）
Atlas 500 Pro 智能边缘服务器（型号 3000）
Atlas 900 AI集群（型号 9000）

2 使用工具

2.1 使用前准备

支持的操作系统

产品	操作系统
A500 Pro-3000	EulerOS 2.8 (默认内核4.19，系统架构aarch64) Ubuntu 18.04 (默认内核4.15，系统架构aarch64) CentOS 7.6（默认内核4.14，系统架构aarch64）
A300-3010	Ubuntu 18.04 (默认内核4.15，系统架构x86_64) CentOS 7.6（默认内核4.14，系统架构x86_64）
A800 9000	EulerOS 2.8 (默认内核4.19，系统架构aarch64) Ubuntu 18.04 (默认内核4.15，系统架构aarch64) CentOS 7.6（默认内核4.14，系统架构aarch64）

产品	操作系统
A800 9010	Ubuntu 18.04(内核 4.15，系统架构x86_64) CentOS 7.6（ 内核4.19.28，系统架构x86_64 ）
A300T 9000	Ubuntu 18.04(内核 4.15，系统架构x86_64) CentOS 7.6（ 内核4.19.28，系统架构x86_64 ）
A800 3000	EulerOS 2.8 (默认内核4.19，系统架构aarch64) Ubuntu 18.04 (默认内核4.15，系统架构aarch64) CentOS 7.6（ 默认内核4.14，系统架构aarch64 ）
A800 3010	Ubuntu 18.04 (默认内核4.15，系统架构x86_64) CentOS 7.6（ 默认内核4.14，系统架构x86_64 ）
A900 9000	EulerOS 2.8 (默认内核4.19，系统架构aarch64) Ubuntu 18.04 (默认内核4.15，系统架构aarch64) CentOS 7.6（ 默认内核4.14，系统架构aarch64 ）

静态服务器使用约束

使用“Ascend-DMI”工具实现以下章节的相关检测时，有如下几点注意事项。

- root用户的默认安装路径为“/usr/local/Ascend”，本手册所有安装路径以此为例，用户可根据实际安装路径修改。
- 添加环境变量，不同用户之间环境变量不共享，需要分别添加，具体操作如下。
  a. 若需要使用指定用户执行ascend-dmi工具，则要求此用户必须已加入CANN软件运行用户属组。若通过root用户安装ascend-dmi工具包，则属组默认为HwHiAiUser；若通过非root用户安装ascend-dmi工具包，则属组默认与该用户相同。可通过执行以下命令查询CANN软件运行用户属组。

     source /etc/ascend_install.info; echo ${UserGroup}

  b. 执行**vi ~/.bashrc**命令。
  c. 根据使用场景不同，将下述环境变量加入“.bashrc”文件，保存退出。
     - 推理

       ```
       toolbox_install_path=/usr/local/Ascend/toolbox/latest
       nnrt_install_path=/usr/local/Ascend/nnrt/latest
       export PATH=${toolbox_install_path}/Ascend-DMI/bin:${PATH}
       export LD_LIBRARY_PATH=/usr/local/dcmi:${toolbox_install_path}/Ascend-DMI/lib64:${nnrt_install_path}/acllib/lib64:/usr/local/Ascend/driver/lib64:${LD_LIBRARY_PATH}
       ```

     - 训练

       ```
       toolbox_install_path=/usr/local/Ascend/toolbox/latest
       nnae_install_path=/usr/local/Ascend/nnae/latest
       export PATH=${toolbox_install_path}/Ascend-DMI/bin:${PATH}
       export LD_LIBRARY_PATH=/usr/local/dcmi:${toolbox_install_path}/Ascend-DMI/lib64:${nnae_install_path}/fwkacllib/lib64:/usr/local/Ascend/driver/lib64/common:/usr/local/Ascend/driver/lib64/driver:${LD_LIBRARY_PATH}
       ```

  d. 执行命令**source ~/.bashrc**，使环境变量生效。
- “Ascend-DMI”工具的检测命令可在任意目录执行，检测命令介绍请见**带宽测试**至**设备拓扑检测**。

- 用户可以**root**用户或非**root**用户使用工具。若以非**root**用户使用工具，需要执行以下步骤加入软件包运行用户属组（若在安装软件包时使用了**--install-for-all**，则无需执行此操作）。例如软件包运行用户属组为HwHiAiUser，操作如下：
  a. 以**root**用户登录服务器。
  b. 执行**usermod -a -G HwHiAiUser *{username}***命令，加入属组。***{username}***为非root用户名，请用户自行替换。

功能使用约束

- Atlas 800 训练服务器（型号 9010）场景功能支持情况如下：

功能	物理机	宿主机+容器	虚拟机
带宽测试	支持	支持	不支持
算力测试	支持	支持	不支持
功耗测试	支持	支持	不支持
设备实时状态查询	支持	支持	不支持
故障诊断	支持	支持	不支持
软硬件版本兼容性测试	支持	支持	不支持
设备拓扑检测	支持	支持	不支持

- Atlas 300T 训练卡（型号 9000）场景功能支持情况如下：

功能	物理机	宿主机+容器	虚拟机
带宽测试	支持	支持	不支持
算力测试	支持	支持	不支持
功耗测试	支持	支持	不支持
设备实时状态查询	支持	支持	不支持
故障诊断	支持	支持	不支持
软硬件版本兼容性测试	支持	支持	不支持
设备拓扑检测	支持	支持	不支持

- Atlas 300I 推理卡（型号 3000）/Atlas 300I 推理卡（型号 3010）/Atlas 800 推理服务器（型号 3000）/Atlas 800 推理服务器（型号 3010）/Atlas 500 Pro 智能边缘服务器（型号 3000）场景功能支持情况如下：

功能	物理机	宿主机+容器	虚拟机
带宽测试	支持	支持	支持
算力测试	支持	支持	支持
功耗测试	支持	不支持	不支持
设备实时状态查询	支持	功耗值为NA，其他正常支持	功耗值为NA，其他正常支持
故障诊断	支持	功耗值压测为NA，其他正常支持	功耗值压测为NA，其他正常支持
软硬件版本兼容性测试	支持	支持	支持
设备拓扑检测	支持	支持	支持

- Atlas 800 训练服务器（型号 9000）、Atlas 900 AI集群（型号 9000）在四种场景（物理机、宿主机+容器、虚拟机）下，所有功能都支持。

容器内使用约束

- 若需要在容器中使用“Ascend-DMI”工具，需将如下文件或文件夹挂载到容器中：

```
/usr/local/dcmi
/usr/local/sbin/npu-smi
/var/log/npu
/usr/slog/slog
/usr/local/Ascend/ascend-toolkit
/usr/local/Ascend/version.info
/usr/local/Ascend/toolbox
/etc/ascend_install.info
```

- 挂载命令举例如下：

  1.适用于EulerOS

```
docker run --cap-add=SYS_ADMIN --rm \
--device=/dev/davinci0 \
--device=/dev/davinci1 \
--device=/dev/davinci2 \
--device=/dev/davinci3 \
--device=/dev/davinci4 \
--device=/dev/davinci5 \
--device=/dev/davinci6 \
--device=/dev/davinci7 \
--device=/dev/davinci_manager \
--device=/dev/hisi_hdc \
--device=/dev/devmm_svm \
-v /usr/local/dcmi:/usr/local/dcmi \
-v /usr/local/sbin/npu-smi:/usr/local/sbin/npu-smi \
-v /var/log/npu:/var/log/npu \
-v /usr/slog/slog:/usr/slog/slog \
-v /usr/local/Ascend/ascend-toolkit:/usr/local/Ascend/ascend-toolkit \
-v /usr/local/Ascend/version.info:/usr/local/Ascend/version.info \
-v /usr/local/Ascend/toolbox:/usr/local/Ascend/toolbox \
-v /etc/ascend_install.info:/etc/ascend_install.info \
-v /usr/local/Ascend/driver:/usr/local/Ascend/driver \
-it ubuntu_arm_b070:18 /bin/bash
```

  2.适用于其他操作系统（已安装ascend-docker-runtime）

```
docker run --cap-add=SYS_ADMIN --rm -e ASCEND_VISIBLE_DEVICES=0-4 \
-v /usr/local/dcmi:/usr/local/dcmi \
```



```
-v /usr/local/sbin/npu-smi:/usr/local/sbin/npu-smi \
-v /var/log/npu:/var/log/npu \
-v /usr/slog/slog:/usr/slog/slog \
-v /usr/local/Ascend/ascend-toolkit:/usr/local/Ascend/ascend-toolkit \
-v /usr/local/Ascend/version.info:/usr/local/Ascend/version.info \
-v /usr/local/Ascend/toolbox:/usr/local/Ascend/toolbox \
-v /etc/ascend_install.info:/etc/ascend_install.info \
-it ubuntu:latest /bin/bash
```

挂载命令中部分内容用户可自行替换，具体如**表2-1**。

表 2-1 挂载命令

挂载命令	说明
-e ASCEND_VISIBLE_DEVICES=0-4	将0号至4号设备挂载到容器中，用户可自行指定可用设备。 使用ASCEND_VISIBLE_DEVICES环境变量指定被挂载至容器中的NPU设备，使用设备序号指定设备。
/usr/local/Ascend/ascend-toolkit:/usr/local/Ascend/ascend-toolkit	root用户安装软件的默认路径，非root用户默认安装目录示例为 /home/xxx/Ascend/ascend-toolkit /home/xxx/Ascend/toolbox xxx为非root用户名。
/usr/local/Ascend/toolbox:/usr/local/Ascend/toolbox	
ubuntu:latest	容器镜像名：标签名，用户可自行替换。

- 挂载完成后，请在容器内参照**不同业务场景的环境变量配置方法**配置环境变量。

2.2 查看帮助信息

查看Ascend-DMI工具帮助信息。

表 2-2 参数说明

参数	说明	是否必填
[-h，--help]	查看Ascend-DMI工具帮助信息。	是

使用实例

ascend-dmi -h

2.3 查看版本信息

查看Ascend-DMI工具版本信息。

表 2-3 参数说明

参数	说明	是否必填
[-v，--version]	查看Ascend-DMI工具版本信息。	是
[-fmt, --format]	指定输出格式，可以为normal或者json，若未指定则默认为normal。	否

使用实例

ascend-dmi -v

2.4 带宽测试

测试项功能

带宽测试主要用于测试总线带宽、内存带宽和时延。

注意事项

为了避免频繁输出日志影响测试结果，测试前确认host和device的日志级别设置为ERROR，确认方法如下。

1. 确认“/var/log/npu/conf/slog/slog.conf”配置文件中的global_level等于3；
2. 如果环境安装了ascend-toolkit组件，以HwHiAiUser用户登录安装ascend-toolkit组件的服务器，执行如下命令查询日志级别，确认GLOBAL_LEVEL和各个子模块的日志级别为ERROR，enableEvent为DISABLE。

   adc --host *<ip>*:*<port>* **--log GetLogLevel**

   ADC命令请参考如下手册：

   推理场景：请参考《CANN V100R020C10 开发辅助工具指南（推理）》的“**设置日志级别**”章节。

   训练场景：请参考《CANN V100R020C10 开发辅助工具指南（训练）》的“**设置日志级别**”章节。

测试项参数查询

用户可任选以下指令之一查看带宽测试命令的可用参数。

ascend-dmi -bw -h

ascend-dmi -bw --help

命令各参数解释如**表2-4**所示。

表 2-4 参数说明

参数	说明	是否必填
[-bw，bandwidth]	使用该参数测试芯片的带宽。	是

参数	说明	是否必填
[-t，--type]	指测试数据流向的分类。 当使用带宽和时延测试功能时，测试的数据流可以分为以下方向，若不填写数据流方向则默认返回h2d、d2h、d2d三个方向的带宽和时延信息。 ● h2d：指数据从Host侧内存通过PCIe总线搬移到Device侧内存，测试整体带宽及时延。 ● d2h：指数据从Device侧内存通过PCIe总线搬移到Host侧内存，测试整体带宽及时延。 ● d2d：指数据从Device侧内存搬移到同一Device侧内存（主要是用于测试Device侧的内存带宽），测试整体带宽及时延。 ● p2p：测试指定源头Device到目标Device的传输速率和时延，仅支持Ascend 910系列芯片设备。	否
[-s，--size]	指传输数据大小并指定测试结果显示方式，单位为字节。 显示方式分为定长模式和步长模式。若“-s”参数缺省则为步长模式，输出传输数据的带宽测试结果，传输数据的范围为2Byte~32M。若“-s”参数不缺省则为定长模式，“-s”参数后必须填写数值指定传输数据的大小，传输数据的取值范围为1Byte~512M，不填写属于错误写法。	否
[-et，--execute-times]	指迭代次数，即内存拷贝次数。取值范围为[1, 1000]，若不填写拷贝次数则默认为5。	否
[-d， --device]	指定需要测试带宽的Device ID，Device ID是指昇腾芯片的ID，用户可以执行**ascend-dmi info**命令，在显示界面表格中的Chip参数处获得芯片数量。比如一个Atlas 300I 推理卡配置4个昇腾芯片，则Device ID的取值范围为[0，3]。若不填写Device ID则默认返回Device 0带宽信息。	否
[-ds，--device-src]	指定p2p测试的源头Device的ID号，若未指定则默认为0。	否
[-dd，--device-dst]	指定p2p测试的目标Device的ID号，若未指定则默认为1。	否

参数	说明	是否必填
[-fmt, --format]	指定输出格式，可以为normal或json。若未指定则默认为normal。	否

说明

-ds与-dd参数需要配合使用，单独使用是错误用法，使用时参数后的数值不能相同。

使用实例

使用实例中命令的回显在推理服务器与训练服务器类似，截图取自推理服务器。需要注意，p2p测试命令截图取自训练服务器，因为p2p测试仅支持Ascend 910系列芯片设备，即训练服务器。

- 以不带参数为例（不带参数则默认查询在Device 0，以h2d、d2h、d2d三个数据流向和步长模式显示的带宽时延信息）。

  ascend-dmi -bw

- 以测试数据从Host侧传输到Device 0，迭代100次的带宽与时延为例。
  - 定长模式。

    ascend-dmi -bw -t h2d -d 0 -s 8388608 -et 100

    若返回如图2-1所示信息，表示工具运行正常，图中参数介绍如表2-5所示。

    图 2-1 带宽测试示例（定长模式）

    ```
    Host to Device Test
    Device  0: Ascend 310.
    ------------------------------------------------------------
     ID  Size(Bytes)  Execute Times  Bandwidth(MB/s)  Latency(us)
    ------------------------------------------------------------
      0     8388608        100            1356.93       5895.70
    ------------------------------------------------------------
    ```

  - 步长模式。

    ascend-dmi -bw -t h2d -d 0 -et 100

    若返回如所图2-2示信息，表示工具运行正常，图中参数介绍如表2-5所示。

图 2-2 带宽测试示例（步长模式）

```
Host to Device Test
Device  0: Ascend 310.
------------------------------------------------------------
 Size(Bytes)  Execute Times  Bandwidth(MB/s)   Latency(us)
------------------------------------------------------------
         2         100            0.18            10.45
         4         100            0.40             9.66
         8         100            0.77             9.94
        16         100            1.51            10.13
        32         100            3.06             9.98
        64         100            6.14             9.94
       128         100           12.32             9.91
       256         100           24.85             9.83
       512         100           51.03             9.57
      1024         100           98.84             9.88
      2048         100          178.50            11.21
      4096         100          290.75            13.44
      8192         100          464.72            16.81
     16384         100          652.16            23.96
     32768         100          810.28            38.57
     65536         100          883.42            70.75
    131072         100          943.90           132.46
    262144         100          979.88           255.14
    524288         100         1147.72           435.65
   1048576         100         1345.56           743.19
   2097152         100         1347.87          1483.88
   4194304         100         1354.96          2952.14
   8388608         100         1356.20          5898.83
  16777216         100         1360.89         11757.05
  33554432         100         1364.74         23447.77
------------------------------------------------------------
```

- 以测试数据从源头Device 1传输到目标Device 2的p2p测试为例。
  ascend-dmi -bw -t p2p -ds 1 -dd 2
  若返回如图2-3所示信息，表示工具运行正常，图中参数介绍如表2-5所示。

图 2-3 p2p 方式带宽测试示例

```
Unidirectional Peer to Peer Test
Device  1 (Ascend 910A) to Device  2 (Ascend 910A)
-----------------------------------------------------------
 Size(Bytes)  Execute Times  Bandwidth(MB/s)   Latency(us)
-----------------------------------------------------------
         2          5            0.01           311.77
         4          5            0.01           310.18
         8          5            0.03           302.08
        16          5            0.05           307.16
        32          5            0.10           309.31
        64          5            0.23           268.06
       128          5            0.39           315.11
       256          5            0.79           307.32
       512          5            1.87           264.25
      1024          5            3.22           303.27
      2048          5            6.95           285.39
      4096          5           12.80           305.26
      8192          5           24.96           313.04
     16384          5           50.46           309.86
     32768          5           99.72           313.52
     65536          5          202.33           308.99
    131072          5          407.03           307.32
    262144          5          937.59           266.71
    524288          5         1848.22           270.61
   1048576          5         3424.25           292.06
   2097152          5         5996.25           333.71
   4194304          5         9348.24           428.12
   8388608          5        13187.25           606.78
  16777216          5        16717.33           957.09
  33554432          5        19038.96          1680.85
-----------------------------------------------------------

Bidirectional Peer to Peer Test
Transfer between Device  1 (Ascend 910A) and Device  2 (Ascend 910A)
-----------------------------------------------------------
 Size(Bytes)  Execute Times  Bandwidth(MB/s)   Latency(us)
-----------------------------------------------------------
         2          5            0.01           324.96
         4          5            0.02           324.17
         8          5            0.05           316.94
        16          5            0.10           318.69
        32          5            0.19           322.42
        64          5            0.41           300.25
       128          5            0.77           318.45
       256          5            1.54           317.18
       512          5            3.09           315.83
      1024          5            6.73           292.54
      2048          5           12.30           317.73
      4096          5           24.51           318.77
      8192          5           48.68           320.99
     16384          5           97.24           321.63
     32768          5          225.61           278.00
     65536          5          418.57           298.82
    131072          5          895.54           281.41
    262144          5         1956.16           255.66
    524288          5         3827.51           261.31
   1048576          5         6658.08           300.57
   2097152          5        11216.70           356.75
   4194304          5        17694.57           452.12
   8388608          5        24757.23           646.35
  16777216          5        31110.60          1028.62
  33554432          5        35815.42          1786.95
-----------------------------------------------------------
```


表 2-5 显示界面参数介绍

参数	说明
Host to Device Test	带宽数据流方向。有以下显示可能： • Host to Device Test • Device to Host Test • Device to Device Test • Unidirectional Peer to Peer Test • Bidirectional Peer to Peer Test
Device X : Ascend XXX	Device X为当前测试的源设备ID，Ascend XXX为芯片类型。若为推理服务器，芯片型号为Ascend 310；若为训练服务器，芯片型号为Ascend 910 A、Ascend 910 B或Ascend 910 Pro A。
ID	Device ID
Size(Bytes)	传输数据大小
Execute Times	迭代次数
Bandwidth(MB/s)	芯片的带宽
Latency(us)	芯片的时延

2.5 算力测试

测试项功能

算力测试通过构造矩阵乘“A(m,k)*B(k,n)”并执行一定次数的方式，根据运算量与执行多次矩阵乘所耗费时间来计算芯片中AI Core的算力值和满算力下芯片的平均功率。

设计的矩阵乘参数如表2-6所示，默认在最大算力模式下运行。

表 2-6 矩阵乘参数

参数	说明	取值
m	A矩阵行	256
k	A矩阵列，B矩阵行	32
n	B矩阵列	128

注意事项

为了避免频繁输出日志影响测试结果，测试前确认host和device的日志级别设置为ERROR，确认方法如下。

1. 确认“/var/log/npu/conf/slog/slog.conf”配置文件中的global_level等于3；
2. 如果环境安装了ascend-toolkit组件，以HwHiAiUser用户登录安装ascend-toolkit组件的服务器，执行如下命令查询日志级别，确认GLOBAL_LEVEL和各个子模块的日志级别为ERROR，enableEvent为DISABLE。

   adc --host *<ip>*:*<port>* **--log GetLogLevel**

   ADC命令请参考如下手册：

   推理场景：请参考《CANN V100R020C10 开发辅助工具指南（推理）》的“**设置日志级别**”章节。

   训练场景：请参考《CANN V100R020C10 开发辅助工具指南（训练）》的“**设置日志级别**”章节。

测试项参数查询

用户可任选以下指令之一查看算力测试命令的可用参数。

ascend-dmi -f -h

ascend-dmi -f --help

各参数解释如**表2-7**所示。

表 2-7 参数说明

参数	说明	是否必填
[-f, flops]	使用该参数测试芯片的算力。	是
[-d, --device]	指定需要测试算力的Device ID，Device ID是指昇腾芯片的ID，用户可以执行**ascend-dmi info**命令，在显示界面表格中的Chip参数处获得芯片数量。比如一个Atlas 300I 推理卡配置4个昇腾芯片，则Device ID的取值范围为[0，3]。若不填写Device ID则默认返回Device 0带宽信息。	否
[-et, --execute-times]	指定芯片上运行矩阵乘法的执行次数，若不填写执行次数则默认为60。 推理场景单位为百万，参数范围为[10，100]。训练场景单位为十万，参数范围为[10，80]。	否
[-fmt, --format]	指定输出格式，可以为normal或json。若未指定则默认为normal。	否

使用实例

推理服务器以测试Device 2，执行次数为6千万的算力为例。

ascend-dmi -f -d 2 -et 60

若返回如**图2-4**所示信息，表示工具运行正常，图中参数介绍如**表2-8**所示。

图 2-4 推理服务器算力测试示例

```
---------------------------------------------------------------------------------------
  Device        Execute Times        Duration(ms)        TFLOPS@FP16        Power(W)
---------------------------------------------------------------------------------------
  2             120,000,000          22589               11.141             12.800(Rated Power)
---------------------------------------------------------------------------------------
```

训练服务器以测试Device 3，执行次数为8百万的算力为例。

ascend-dmi -f -d 3 -et 80

若返回如图2-5所示信息，表示工具运行正常，图中参数介绍如表2-8所示。

图 2-5 训练服务器算力测试示例

```
---------------------------------------------------------------------------------------
  Device        Execute Times        Duration(ms)        TFLOPS@FP16        Power(W)
---------------------------------------------------------------------------------------
  3             256,000,000          2049                262.016            245.106
---------------------------------------------------------------------------------------
```

表 2-8 显示界面参数说明

参数	说明
Device	Device ID。
Execute Times	单个芯片实际运算中执行矩阵乘的次数。
Duration(ms)	执行多次矩阵乘所耗费的时间。
TFLOPS@FP16	Fp16数据进行算力测试得到的算力值。
Power(W)	满算力下芯片的平均功率，推理服务器的芯片显示的为额定功率。

说明

为保证返回检测结果的正确性和准确性，算力测试需要单独执行。

图2-4和图2-5中的Execute Times数值解释如下：

推理服务器的芯片有2个AI Core，Execute Times为执行矩阵乘的次数（6千万）乘以AI Core的个数（2个），结果为120,000,000。（训练服务器的芯片有32个AI Core，同理可计算训练服务器场景下的Execute Times。）

2.6 功耗测试

测试项功能

功耗测试是通过运行单算子模型来检测整卡或芯片的功耗信息。

注意事项

为了避免频繁输出日志影响测试结果，测试前确认host和device的日志级别设置为ERROR，确认方法如下。

1. 确认“/var/log/npu/conf/slog/slog.conf”配置文件中的global_level等于3；
2. 如果环境安装了ascend-toolkit组件，以HwHiAiUser用户登录安装ascend-toolkit组件的服务器，执行如下命令查询日志级别，确认GLOBAL_LEVEL和各个子模块的日志级别为ERROR，enableEvent为DISABLE。

   adc --host *<ip>*:*<port>* **--log GetLogLevel**

   ADC命令请参考如下手册：

   推理场景：请参考《CANN V100R020C10 开发辅助工具指南（推理）》的“**设置日志级别**”章节。

   训练场景：请参考《CANN V100R020C10 开发辅助工具指南（训练）》的“**设置日志级别**”章节。

测试项参数查询

用户可任选以下指令之一查看功耗测试命令的可用参数。

ascend-dmi -p -h

ascend-dmi -p --help

各参数解释如**表2-9**所示。

表 2-9 参数说明

参数	说明	是否必填
[-p, power]	使用该参数进行整卡或芯片的功耗测试。	是
[-dur, --duration]	指运行时间，若不填写运行时间则默认为600。 单位为秒，取值范围为[60， 604800]。	否
[-it, --interval-times]	指屏幕信息打印刷新的间隔时间，若不填写间隔时间则默认为5。 单位为秒，取值范围为[1, 5]。	否
[-pm, --print-mode]	屏幕输出的打印模式，若不填写打印模式则默认为refresh。 打印模式： • refresh：每次打印清除历史打印信息。 • history：打印保存历史信息。	否

说明

为保证返回检测结果的正确性和准确性，功耗测试需要单独执行。

从运行成本考虑，功耗测试打印次数不一定与理论值相同。以功耗工具运行时间为60s，信息打印刷新的间隔为5s为例，理论上打印次数应为12次，实际次数会低于这个数值。

使用实例

以执行时间为60s，信息的打印间隔信息为5s，屏幕的输出模式为清除历史记录为例。

ascend-dmi -p -dur 60 -it 5 -pm refresh

若推理服务器返回如图2-6所示信息或训练服务器返回如图2-7所示信息，表示工具运行正常，图中参数介绍如表2-10和所示。

图 2-6 功耗测试示例（推理服务器）

```
|=======================+==================+======================|
| Card    Type          | NPU Count        | Power                |
+-----------------------+------------------+----------------------+
| Chip    Name          | Health           | Temperature          |
| Device  ID            | AI Core Usage    | Voltage              |
|=======================+==================+======================|
| 0       Atlas300-3000 | 4                | 48.0W                |
+-----------------------+------------------+----------------------+
|     0   Ascend310     | OK               | 63C                  |
|     0                 | 100%             | 0.80V                |
+-----------------------+------------------+----------------------+
|     1   Ascend310     | OK               | 63C                  |
|     1                 | 100%             | 0.80V                |
+-----------------------+------------------+----------------------+
|     2   Ascend310     | OK               | 63C                  |
|     2                 | 100%             | 0.80V                |
+-----------------------+------------------+----------------------+
|     3   Ascend310     | OK               | 64C                  |
|     3                 | 100%             | 0.80V                |
|=======================+==================+======================|
```

图 2-7 功耗测试示例（训练服务器）

```
|=======================+==================+======================|
| Server Type           | NPU Count                               |
+-----------------------+------------------+----------------------+
| Device ID             | Health           | Temperature   Voltage |
| Chip Name             | AI Core Usage    | Power                |
|=======================+==================+======================|
| Atlas800-9010         | 8                                       |
+-----------------------+------------------+----------------------+
| 0                     | OK               | 75C           12.07V |
| Ascend910A            | 100%             | 263.2W               |
+-----------------------+------------------+----------------------+
| 1                     | OK               | 77C           12.05V |
| Ascend910A            | 100%             | 255.6W               |
+-----------------------+------------------+----------------------+
| 2                     | OK               | 62C           12.03V |
| Ascend910A            | 100%             | 257.5W               |
+-----------------------+------------------+----------------------+
| 3                     | OK               | 76C           12.03V |
| Ascend910A            | 100%             | 252.7W               |
+-----------------------+------------------+----------------------+
| 4                     | OK               | 73C           12.05V |
| Ascend910A            | 100%             | 280.1W               |
+-----------------------+------------------+----------------------+
| 5                     | OK               | 66C           12.06V |
| Ascend910A            | 100%             | 249.8W               |
+-----------------------+------------------+----------------------+
| 6                     | OK               | 67C           12.03V |
| Ascend910A            | 100%             | 266.1W               |
+-----------------------+------------------+----------------------+
| 7                     | OK               | 76C           12.04V |
| Ascend910A            | 100%             | 254.7W               |
|=======================+==================+======================|
```


表 2-10 显示界面参数说明

参数	说明	产品形态
Card	卡编号	标卡
Type	卡类型	
Chip	芯片编号	
Name	芯片名称	
Server Type	服务器类型	训练服务器
Chip Name	芯片名称	
NPU Count	NPU的个数	标卡、训练服务器
Power	当前整卡或芯片的实际功耗	
Health	芯片健康程度	
Temperature	芯片当前温度	
Device ID	芯片设备号	
AI Core Usage	芯片AI Core的使用率	
Voltage	芯片当前电压	

2.7 设备实时状态查询

测试项功能

设备实时状态查询是检测设备在运行过程中的状态信息。

测试项参数查询

用户可任选以下指令之一查看设备实时状态查询命令的可用参数。

ascend-dmi -i -h

ascend-dmi -i --help

各参数解释如表2-11所示。

表 2-11 参数说明

参数	说明	是否必填
[-i，info]	使用该参数进行设备实时状态查询。	是
[-b，--brief]	使用该参数查看芯片的基本信息。	否
[-dt，--detail]	使用该参数查看芯片的详细信息。	否

参数	说明	是否必填
不填写-dt与-b参数	默认查看芯片的基本信息。	否
[-fmt, --format]	指定输出格式，可以为normal或json。若未指定则默认为normal。	否

使用实例

- 以查看芯片的详细信息为例。

  ascend-dmi -i -dt

  若推理服务器返回如图2-8所示信息或训练服务器返回如图2-9所示信息，表示工具运行正常，图中参数介绍如表2-12和所示。

  图 2-8 设备实时状态查询示例（推理服务器）

```
==========Product Details==========
ascend-dmi                          : 20.0.0

Card Quantity                       : 4
    Card Name                       : Atlas300-3000
    Card Manufacturer               : Huawei
    Card Serial Number              : 033EFS10L5000389
    Card ID                         : 0
    Real-time Card Power (W)        : 15.2
    Device Count                    : 4
        Chip Name                   : Ascend310
        Device ID                   : 0
        Chip ID                     : 0
        DIE ID                      : 17c56594-2d04d24-4c096a53-100a8c0-3310003f
        AI Core Information
            AI Core Count           : 2
            AI Core Usage (%)       : 0
            Cube Count              : 2
            Vector Count            : 2
        CPU Information
            AI CPU Count            : 4
            AI CPU Usage (%)        : 0
            Control CPU Count       : 4
            Control CPU Usage (%)   : 0
            Control CPU Frequency (MHz)  : 1600
        Memory Information
            Total (MB)              : 8192
            Used (MB)               : 2539
            Bandwidth Usage (%)     : 0
            Frequency (MHz)         : 1600
        Power Information
            Rated Power (W)         : 12.8
        Temperature (C)             : 47.0
        PCIe Information
            Domain                  : 0x0000
            Bus                     : 0x01
            Device                  : 0x00
            Bus ID                  : 0000:01:00.0
            Subvendor ID            : 0x0200
            Subdevice ID            : 0x0100
            LnkCap Speed            : 8.0GT/s
            LnkCap Width            : 4x
            LnkSta Speed            : 8.0GT/s
            LnkSta Width            : 2x
            CPU Affinity            : 0-31
        Error Information
        Error Count                 : 0
        ECC Information
            DDR                     : Enabled
                Single-Bit Error Count   : 0
                Double-Bit Error Count   : 0
```


图 2-9 设备实时状态查询示例（训练服务器）

```
==============================Product Details==============================
ascend-dmi                                :

Server Type                               : Atlas800-9010
NPU Count                                 : 8
    Chip Name                             : Ascend910A
    Device ID                             : 0
    Chip ID                               : 0
    DIE ID                                : 25e10dd4-22105120-9a1330d4-14cc040a-89102003
    AI Core Information
        AI Core Count                     : 32
        AI Core Usage (%)                 : 0
        Cube Count                        : 32
        Vector Count                      : 32
    CPU Information
        AI CPU Count                      : 16
        AI CPU Usage (%)                  : 0
        Control CPU Count                 : 0
        Control CPU Usage (%)             : 0
        Control CPU Frequency (MHz)       : 2000
    Memory Information
        Total (MB)                        : 15307
        Used (MB)                         : 153
        Bandwidth Usage (%)               : 0
        Frequency (MHz)                   : 1200
    Power Information
        Real-time Power (W)               : 73.4
        Rated Power (W)                   : 284.5
    Temperature (C)                       : 47.0
    PCIe Information
        Domain                            : 0x0000
        Bus                               : 0x61
        Device                            : 0x00
        Bus ID                            : 0000:61:00.0
        Subvendor ID                      : 0x0200
        Subdevice ID                      : 0x0100
        LnkCap Speed                      : 16.0GT/s
        LnkCap Width                      : 16x
        LnkSta Speed                      : 8.0GT/s
        LnkSta Width                      : 16x
        CPU Affinity                      : 0-19,40-59
    Error Information
    Error Count                           : 0
    ECC Information
        DDR                               : Enabled
            Single-Bit Error Count        : 0
            Double-Bit Error Count        : 0
        HBM                               : Enabled
            Single-Bit Error Count        : 0
            Double-Bit Error Count        : 0
```

表 2-12 显示界面参数说明

参数	说明	产品形态
Server Type	服务器类型	训练服务器
NPU Count	NPU数量	
Chip Name	芯片名称	
Card Quantity	卡数量	标卡
Card Name	卡名称	
Card Manufacturer	卡生产商	
Card Serial Number	卡序列号	
Card ID	卡ID号	

参数	说明	产品形态
Real-time Card Power (W)	卡实时功耗	
Device Count	设备数（NPU个数）	
Chip Name	芯片名称	标卡、训练服务器
Device ID	设备ID号	
Chip ID	芯片ID号	
DIE ID	芯片的DIE ID	
AI Core Information	AI Core信息 包括以下信息： ● AI Core Count：AI Core 个数 ● AI Core Usage (%)：AI Core利用率 ● Cube Count：Cube个数 ● Vector Count：Vector个数	
CPU Information	CPU信息 包括以下信息： ● AI CPU Count：AI CPU个数 ● AI CPU Usage (%)：AI CPU利用率 ● Control CPU Count：Control CPU 个数 ● Control CPU Usage (%)：Control CPU利用率 ● Control CPU Frequency (MHz)：Control CPU频率	
Memory Information	内存信息 包括以下信息： ● Total (MB)：总容量 ● Used (MB)：已使用 ● Bandwidth Usage (%)：内存带宽使用率 ● Frequency (MHz)：内存频率	
Power Information	功耗信息 包括以下信息： ● Real-time Power (W)：实时功耗（只有在训练服务器执行命令时显示） ● Rated Power (W)：芯片额定功耗	

参数	说明	产品形态
Temperature (C)	芯片温度	
PCIe Information	PCIe信息 包括以下信息： ● Domain：PCIe域 ● Bus：PCIe总线编号 ● Device：PCIe设备号 ● Bus ID：PCIe总线地址 ● Subvendor ID：子系统厂商识别码 ● Subdevice ID：子设备号 ● LnkCap Speed：链路最大速率 ● LnkCap Width：链路最大带宽 ● LnkSta Speed：链路当前速率 ● LnkSta Width：链路当前带宽 ● CPU Affinity：亲和性CPU	
Error Information	错误信息	
Error Count	错误个数	
ECC Information	ECC信息	
DDR	卡的内存类型，有以下四种内存类型显示可能： ● DDR ● SRAM ● HBM ● NPU 本项显示中包括以下信息： ● Single-Bit Error Count：单bit错误计数 ● Double-Bit Error Count：双bit错误计数	

说明

非**root**用户执行**ascend-dmi -i -dt**命令时，部分检测项会出现<Access denied. Please switch to root and try again.>字样，若用户需要获取这部分信息可切换到**root**用户重新执行命令。

- 以查看芯片的基本信息为例。

  ascend-dmi -i -b

若推理服务器返回如图2-10所示信息或训练服务器返回如图2-11所示信息，表示工具运行正常，图中参数介绍如表2-13所示。

图 2-10 设备实时状态查询示例（推理服务器）

```
+---------------------+----------------------------------+---------------------+
| ascend-dmi      20.0.0                          Brief Information              |
+=====================+==================================+=====================+
| Card    Type        | NPU Count                        | Real-time Card Power |
+---------------------+----------------------------------+---------------------+
| Chip    Name        | Health          Used Memory      | Temperature  Voltage |
| Device ID           | Bus ID          AI Core Usage    |                      |
+=====================+==================================+=====================+
| 69      Atlas300-3010 | 4                              | 25.4W                |
+---------------------+----------------------------------+---------------------+
|    0    Ascend310   | OK              2457MB/8192MB    | 49C          0.80V   |
|    0                | 0000:47:00.0    0%               |                      |
+---------------------+----------------------------------+---------------------+
|    1    Ascend310   | OK              2457MB/8192MB    | 54C          0.83V   |
|    1                | 0000:48:00.0    0%               |                      |
+---------------------+----------------------------------+---------------------+
|    2    Ascend310   | OK              2457MB/8192MB    | 62C          0.83V   |
|    2                | 0000:49:00.0    0%               |                      |
+---------------------+----------------------------------+---------------------+
|    3    Ascend310   | OK              2457MB/8192MB    | 56C          0.83V   |
|    3                | 0000:4A:00.0    0%               |                      |
+=====================+==================================+=====================+
```

图 2-11 设备实时状态查询示例（训练服务器）

```
+---------------------+----------------------------------+---------------------+
| ascend-dmi                                      Brief Information              |
+=====================+==================================+=====================+
| Server Type         | NPU Count                        |                      |
+---------------------+----------------------------------+---------------------+
| Device ID           | Health          Used Memory      | Temperature  Voltage |
| Chip Name           | Bus ID          AI Core Usage    | Power                |
+=====================+==================================+=====================+
| Atlas800-9010       | 8                                |                      |
+---------------------+----------------------------------+---------------------+
| 0                   | OK              153MB/15307MB    | 47C          12.20V  |
| Ascend910A          | 0000:61:00.0    0%               | 73.6W                |
+---------------------+----------------------------------+---------------------+
| 1                   | OK              313MB/15689MB    | 40C          12.20V  |
| Ascend910A          | 0000:DB:00.0    0%               | 70.2W                |
+---------------------+----------------------------------+---------------------+
| 2                   | OK              627MB/15689MB    | 38C          12.20V  |
| Ascend910A          | 0000:B2:00.0    0%               | 71.5W                |
+---------------------+----------------------------------+---------------------+
| 3                   | OK              3120MB/15601MB   | 46C          12.19V  |
| Ascend910A          | 0000:3E:00.0    0%               | 69.5W                |
+---------------------+----------------------------------+---------------------+
| 4                   | OK              306MB/15307MB    | 48C          12.21V  |
| Ascend910A          | 0000:60:00.0    0%               | 74.3W                |
+---------------------+----------------------------------+---------------------+
| 5                   | OK              2196MB/15689MB   | 40C          12.22V  |
| Ascend910A          | 0000:DA:00.0    0%               | 68.7W                |
+---------------------+----------------------------------+---------------------+
| 6                   | OK              627MB/15689MB    | 42C          12.20V  |
| Ascend910A          | 0000:B1:00.0    0%               | 70.2W                |
+---------------------+----------------------------------+---------------------+
| 7                   | OK              1404MB/15601MB   | 51C          12.21V  |
| Ascend910A          | 0000:3D:00.0    0%               | 70.8W                |
+=====================+==================================+=====================+
```

表 2-13 显示界面参数说明

参数	说明	产品形态
Card	卡编号	标卡
Type	卡类型	
NPU Count	NPU的个数	

参数	说明	产品形态
Real-time Card Power	当前板卡的实际功耗	
Chip	芯片编号	
Name	芯片名称	
Server Type	服务器类型	训练服务器
NPU Count	NPU数量	
Chip Name	芯片名称	
Power	功耗	
Health	芯片健康程度	标卡、训练服务器
Used Memory	内存使用信息	
Temperature	芯片当前温度	
Voltage	芯片当前电压	
Device ID	芯片设备号	
Bus ID	芯片的Bus ID	
AI Core Usage	芯片AI Core的使用率	

2.8 故障诊断

测试项功能

故障诊断会获取芯片健康信息，同时对芯片进行算力、功耗、带宽测试并输出测试结果，用以判断当前产品的健康状态。

测试项参数查询

用户可任选以下指令之一查看故障诊断命令的可用参数。

ascend-dmi -dg -h

ascend-dmi -dg --help

各参数解释如**表2-14**所示。

表 2-14 参数说明

参数	说明	是否必填
[-dg, diagnosis]	使用该参数进行整卡的故障诊断测试。	是

参数	说明	是否必填
[-c, --card]	指定Card进行诊断，不指定则默认诊断所有Card的信息。 用户可以执行**ascend-dmi info**命令，在显示界面表格中的Card参数处获得已安装的卡的号码。	否
[-l, --level]	指定诊断等级，不指定则默认诊断Level 0等级信息。 目前支持以下两种等级： • 0：诊断芯片的健康信息。 • 1：诊断卡的功耗、带宽、算力等信息。	否
不填写-c与-l参数	诊断所有card的Level 0等级信息。	否
[-fmt, --format]	指定输出格式，可以为normal或json。若未指定则默认为normal。	否

说明

为保证返回检测结果的正确性和准确性，故障诊断需要单独执行。

使用实例

使用实例中命令的回显在推理服务器与训练服务器类似，截图取自推理服务器。

- 以指定卡号和诊断等级Level 0为例。

  ascend-dmi -dg -c 65 -l 0

  65为环境中已安装的卡号码的示例，用户可以执行**ascend-dmi info**命令，在显示界面表格中的Card参数处获得已安装的卡的号码，请用户自行替换。

  若返回如图2-12所示信息，表示工具运行正常，图中参数介绍如表2-15所示。

  图 2-12 故障诊断示例（ Level 0 ）

  ```
  Card                                : 65
    Device                            : 4
      Health                          : OK

    Device                            : 5
      Health                          : OK

    Device                            : 6
      Health                          : OK

    Device                            : 7
      Health                          : OK
  ```

- 以指定卡号和诊断等级Level 1为例。

  ascend-dmi -dg -c 65 -l 1

65为环境中已安装的卡号码的示例，用户可以执行**ascend-dmi info**命令，在显示界面表格中的Card参数处获得已安装的卡的号码，请用户自行替换。

若返回如**图2-13**所示信息，表示工具运行正常，图中参数介绍如**表2-15**所示。

图 2-13 故障诊断示例（Level 1）

```
Card                              :    65
  Device                          :    4
    Health                        :    OK
    Flops Test
      Duration(ms)                :    3766
      Computing Power(TFLOPS)     :    11.137291
    Bandwidth Test                :    Device To Device
      Bandwidth(MB/s)             :    45397.267701
    Bandwidth Test                :    Device To Host
      Bandwidth(MB/s)             :    2936.311216
    Bandwidth Test                :    Host To Device
      Bandwidth(MB/s)             :    2557.242812

  Device                          :    5
    Health                        :    OK
    Flops Test
      Duration(ms)                :    3766
      Computing Power(TFLOPS)     :    11.137291
    Bandwidth Test                :    Device To Device
      Bandwidth(MB/s)             :    45483.661086
    Bandwidth Test                :    Device To Host
      Bandwidth(MB/s)             :    2921.620144
    Bandwidth Test                :    Host To Device
      Bandwidth(MB/s)             :    2571.231763

  Device                          :    6
    Health                        :    OK
    Flops Test
      Duration(ms)                :    3766
      Computing Power(TFLOPS)     :    11.137291
    Bandwidth Test                :    Device To Device
      Bandwidth(MB/s)             :    45477.239995
    Bandwidth Test                :    Device To Host
      Bandwidth(MB/s)             :    2928.073737
    Bandwidth Test                :    Host To Device
      Bandwidth(MB/s)             :    2525.945231

  Device                          :    7
    Health                        :    OK
    Flops Test
      Duration(ms)                :    3766
      Computing Power(TFLOPS)     :    11.137291
    Bandwidth Test                :    Device To Device
      Bandwidth(MB/s)             :    45415.762753
    Bandwidth Test                :    Device To Host
      Bandwidth(MB/s)             :    2884.494700
    Bandwidth Test                :    Host To Device
      Bandwidth(MB/s)             :    2512.327214

  Power Test
    Max Power(W)                  :    62.600002
    Average Power(W)              :    60.729111
    Max AI Core(%)                :    100
    Average AI Core(%)            :    89
    Max Temp(C)                   :    83
    Average Temp(C)               :    80
    Max Voltage(V)                :    75
    Average Voltage(V)            :    75
```


表 2-15 实现界面参数说明

参数	说明
Card	卡号
Device	芯片编号
Health	芯片的健康程度
Flops Test	算力工具测试
Duration(ms)	测试时间
Computing Power(TFLOPS)	算力
Bandwidth Test	带宽测试类型
Bandwidth(MB/s)	带宽值
Power Test	功耗测试
Max Power(W)	最大功耗
Average Power(W)	平均功耗
Max AI Core(%)	AI Core最大利用率
Average AI Core(%)	AI Core平均利用率
Max Temp(C)	最大温度
Average Temp(C)	平均温度
Max Voltage(V)	最大电压
Average Voltage(V)	平均电压

2.9 软硬件版本兼容性测试

测试项功能

软硬件兼容性工具会获取硬件信息、架构、驱动版本、固件版本以及软件版本，并检测软硬件间的兼容性。

使用限制

固件版本只有在用root用户执行ascend-dmi工具的时候才可以查询。

测试项参数查询

用户可任选以下指令之一查看软硬件版本兼容性测试命令的可用参数。

ascend-dmi -c -h

ascend-dmi -c --help

各参数解释如表2-16所示。

表 2-16 参数说明

参数	说明	是否必填
[-c, compatible]	使用该参数进行软硬件版本兼容性检测。	是
[-p, --path]	用户指定检测兼容性的软件包的安装路径。 root用户安装软件包时采用的是默认路径“/usr/local/Ascend”，可不填写此参数。 若软件包安装在用户指定目录下，用户使用兼容性检测功能时，需提供软件包安装目录。例如软件包安装在“/home/xxx/Ascend”目录下，指令为： **ascend-dmi -c -p /home/xxx/Ascend**	否
[-fmt, --format]	指定输出格式，可以为normal或json。若未指定则默认为normal。	否

说明

兼容性工具检测的软件包如下：

- toolkit：开发套件
- nnrt：离线推理引擎
- nnae：训练/在线推理软件包
- tfplugin：框架插件
- toolbox：实用工具
- npu-driver：驱动
- npu-firmware：固件

使用实例

以测试软硬件版本兼容性为例。

ascend-dmi -c

若推理服务器返回如图2-14所示信息或训练服务器返回如图2-15所示信息，表示工具运行正常，图中参数请参见表2-17。

图 2-14 软硬件版本兼容性检测示例（推理服务器）

```
|==========================================================================================|
|                                   System Information                                     |
|==========================================================================================|
| Architecture        | x86_64                                                             |
+---------------------+--------------------------------------------------------------------+
| Product Type        | Atlas300-3010                                                      |
|==========================================================================================|
|                          Compatibility Check Result: Compatible                          |
|==========================================================================================|
| Package             | Version            | Status                     | Dependencies     |
+---------------------+--------------------+----------------------------+------------------+
| npu-driver          | 20.1.0             | OK                         | NA               |
+---------------------+--------------------+----------------------------+------------------+
| npu-firmware        | 1.75.22.1.220      | OK                         | NA               |
+---------------------+--------------------+----------------------------+------------------+
| nnrt                | 20.1.spc200        | OK                         | NA               |
+---------------------+--------------------+----------------------------+------------------+
| toolbox             | 20.1.spc200        | OK                         | NA               |
|==========================================================================================|
```

图 2-15 软硬件版本兼容性检测示例（训练服务器）

```
|==========================================================================================|
|                                   System Information                                     |
|==========================================================================================|
| Architecture        | x86_64                                                             |
+---------------------+--------------------------------------------------------------------+
| Product Type        | Atlas800-9010                                                      |
|==========================================================================================|
|                          Compatibility Check Result: Compatible                          |
|==========================================================================================|
| Package             | Version            | Status                     | Dependencies     |
+---------------------+--------------------+----------------------------+------------------+
| npu-driver          | 20.1.0             | OK                         | NA               |
+---------------------+--------------------+----------------------------+------------------+
| npu-firmware        | 1.75.22.1.220      | OK                         | NA               |
+---------------------+--------------------+----------------------------+------------------+
| nnae                | 20.1.0             | OK                         | NA               |
+---------------------+--------------------+----------------------------+------------------+
| toolbox             | 20.1.0             | OK                         | NA               |
|==========================================================================================|
```

表 2-17 显示界面参数说明

参数	说明
System Information	系统信息
Architecture	架构
Product Type	硬件类型
Compatibility Check Result	兼容性检测结果
Package	包名
Version	版本
Status	状态
Dependencies	依赖

2.10 设备拓扑检测

测试项功能

设备拓扑检测是查询单机内多卡间的拓扑结构。

测试项参数查询

用户可任选以下指令之一查看设备拓扑检测命令的可用参数。

ascend-dmi -topo -h

ascend-dmi -topo --help

各参数解释如表2-18所示。

表 2-18 参数说明

参数	说明	是否必填
[-topo, topology]	查看单机内多卡间的拓扑结构。	是
[-fmt, --format]	指定输出格式，可以为normal或json。若未指定则默认为normal。	否

使用实例

使用实例中命令的回显在推理服务器与训练服务器上类似，截图取自推理服务器。

- 以查看拓扑结构为例。
  ascend-dmi -topo
  若返回如图2-16所示信息，表示工具运行正常，图中参数请参见表2-19。

  图 2-16 设备拓扑检测示例

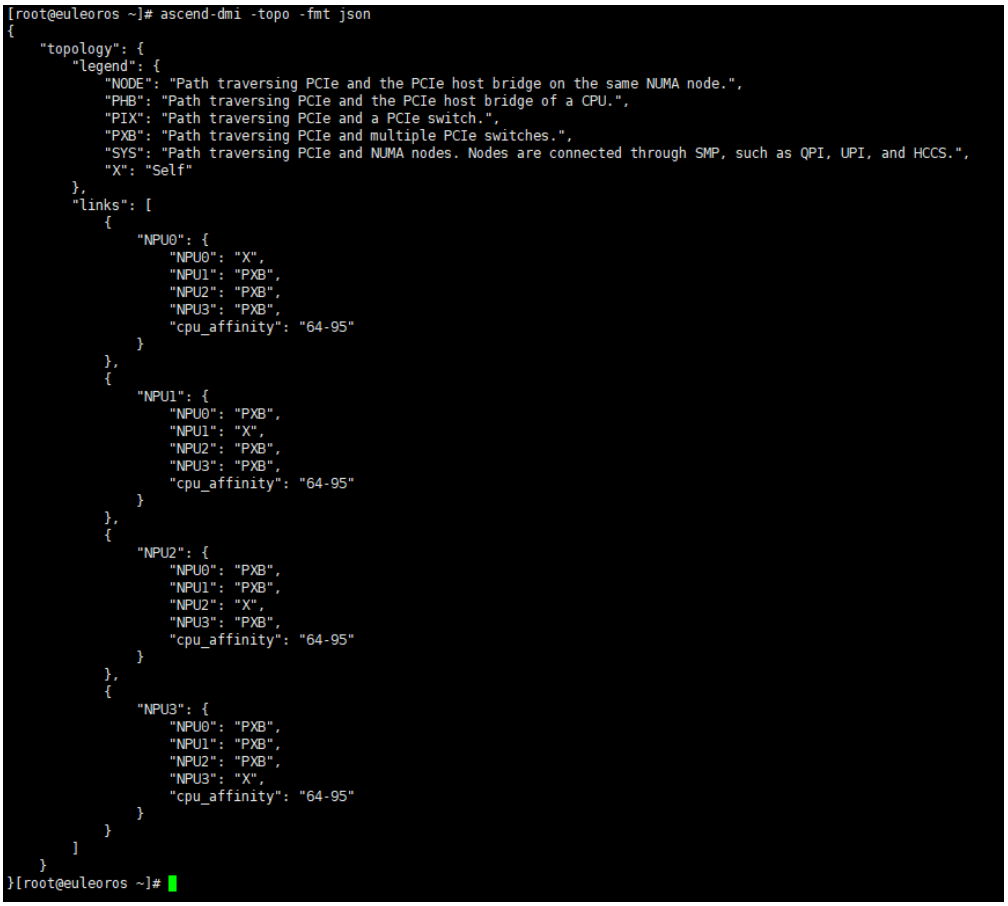
```
        NPU0    NPU1    NPU2    NPU3    CPU Affinity
NPU0     X      PIX     PIX     PIX     0-3,8-11
NPU1    PIX      X      PIX     PIX     0-3,8-11
NPU2    PIX     PIX      X      PIX     0-3,8-11
NPU3    PIX     PIX     PIX      X      0-3,8-11

Legend:

  X    = Self
  SYS  = Path traversing PCIe and NUMA nodes. Nodes are connected through SMP, such as QPI, UPI, and HCCS.
  NODE = Path traversing PCIe and the PCIe host bridge on the same NUMA node.
  PHB  = Path traversing PCIe and the PCIe host bridge of a CPU.
  PXB  = Path traversing PCIe and multiple PCIe switches.
  PIX  = Path traversing PCIe and a PCIe switch.
```

- 以指定输出格式为json为例
  ascend-dmi -topo -fmt json
  若返回如图2-17所示信息，表示工具运行正常，图中参数请参见表2-19。

图 2-17 设备拓扑检测 json 输出示例



```
[root@euleoros ~]# ascend-dmi -topo -fmt json
{
    "topology": {
        "legend": {
            "NODE": "Path traversing PCIe and the PCIe host bridge on the same NUMA node.",
            "PHB": "Path traversing PCIe and the PCIe host bridge of a CPU.",
            "PIX": "Path traversing PCIe and a PCIe switch.",
            "PXB": "Path traversing PCIe and multiple PCIe switches.",
            "SYS": "Path traversing PCIe and NUMA nodes. Nodes are connected through SMP, such as QPI, UPI, and HCCS.",
            "X": "Self"
        },
        "links": [
            {
                "NPU0": {
                    "NPU0": "X",
                    "NPU1": "PXB",
                    "NPU2": "PXB",
                    "NPU3": "PXB",
                    "cpu_affinity": "64-95"
                }
            },
            {
                "NPU1": {
                    "NPU0": "PXB",
                    "NPU1": "X",
                    "NPU2": "PXB",
                    "NPU3": "PXB",
                    "cpu_affinity": "64-95"
                }
            },
            {
                "NPU2": {
                    "NPU0": "PXB",
                    "NPU1": "PXB",
                    "NPU2": "X",
                    "NPU3": "PXB",
                    "cpu_affinity": "64-95"
                }
            },
            {
                "NPU3": {
                    "NPU0": "PXB",
                    "NPU1": "PXB",
                    "NPU2": "PXB",
                    "NPU3": "X",
                    "cpu_affinity": "64-95"
                }
            }
        ]
    }
}[root@euleoros ~]#
```

表 2-19 显示界面参数说明

参数	说明
NPUx	服务器安装的NPU，x为NPU的ID。
SYS	通过PCIe连接且穿过NUMA nodes，nodes之间通过SMP连接，如：QPI、UPI和HCCS。
NODE	通过PCIe连接且穿过PCIe host bridge，但是在同个NUMA node。
PHB	通过PCIe连接且穿过同一个CPU的PCIe host bridge。
PXB	通过PCIe连接且穿过多个PCIe switch。
PIX	通过PCIe连接且穿过同一个PCIe switch。
CPU Affinity	一个NUMA node下的Device和CPU具有亲和性。

说明

以图2-16中的CPU Affinity为例，0-3表示CPU0到CPU3，8-11表示CPU8到CPU11，NPU和以上八个CPU都具有亲和性。

3 日志收集

3.1 简介

ascend-log-collect.sh用于在故障分析定位时收集运行环境信息、昇腾NPU健康信息和昇腾软件日志。收集到的数据以tar.gz格式保存。

说明

日志收集功能收集的日志可能包含系统信息，请用户注意日志导出后使用过程中的信息扩散风险。

- 运行环境信息包含以下内容：
  - 操作系统信息，通过读取“/etc/*release”文件获取。
  - PCIe设备信息，通过执行**lspci**命令获取。
  - 系统软件包信息，通过执行**apt list/rpm -qa**命令获取。
  - Python软件包信息，通过执行**pip list/pip3 list**命令获取。
- 昇腾NPU健康信息包含以下内容：
  - 昇腾软件安装信息。
  - Ascend-DMI版本信息、硬件检测、拓扑检测、level0诊断、level1诊断（extra模式）、功耗测试（extra模式）等结果，通过执行**ascend-dmi**相关命令获取。
  - 芯片相关信息，通过执行**npu-smi**相关命令获取。
- 收集的昇腾软件日志包含以下目录：
  - “/var/dlogshichak”
  - “/var/log/npu/bbox”
  - “/var/log/npu/conf”
  - “/var/log/npu/hisi_logs”

- “/var/log/npu/ide_aemon”
- “/var/log/npu/oplog”
- “/var/log/npu/slog”
- “/var/log/npu/toolchain”
- “/var/log/ascend-dmi”（**root**帐户。不推荐。）
- “${HOME}/var/log/ascend-dmi”（普通帐户）

3.2 使用前准备

- Ascend-DMI已正确配置环境变量。若未配置，请参考**使用前准备**进行配置。
- 存放输出结果的目录的剩余空间至少是“/var/log/npu”目录已使用空间的1.5倍。

3.3 约束

需要使用昇腾软件安装帐户执行命令**ascend-log-collect.sh**，默认帐户为**HwHiAiUser**。不推荐使用**root**帐户执行。

如需查询昇腾软件安装帐户，请执行命令**source /etc/ascend_install.info; echo ${UserName}**，如图3-1所示。

图 3-1 查询昇腾软件安装帐户

```
[root@localhost ~]# source /etc/ascend_install.info; echo ${UserName}
HwHiAiUser
```

3.4 使用方式

3.4.1 查看帮助信息

查看ascend-log-collect.sh脚本帮助信息。

表 3-1 参数说明

参数	说明	是否必填
[-h，--help]	查看ascend-log-collect.sh脚本帮助信息。	是

使用实例

ascend-log-collect.sh --help


```
[root@euleoros ~]# ascend-log-collect.sh --help
ascend-log-collect.sh is used to collect operating environment information, Ascend
NPU health information, and Ascend software logs during fault analysis and locating.
The collected data is saved in tar.gz format.

Command: ascend-log-collect.sh [OPTIONS]...

Options:
  -h, --help                          Displays the help information.
  -v, --version                       Displays the version information.
  --output-file=<FILENAME>            Specifies the name of the output file. The file
                                      name extension .tar.gz is recommended.
                                      [ascend-report-<hostname>-<YYYYMMDDhhmmss>.tar.gz]
  --safe                              Calls ascend-dmi to perform Ascend NPU diagnosis
                                      in level 0 mode. The power consumption test is not
                                      performed at this time. For more information about
                                      ascend-dmi, run ascend-dmi -h.
                                      [This option is enabled by default.]
  --extra                             Calls ascend-dmi to perform Ascend NPU diagnosis
                                      in level 1 mode. If the power consumption test is
                                      performed, the AI core usage and chip temperature
                                      increase. After the collection is complete, the AI
                                      core usage and chip temperature become normal. For
                                      more information about ascend-dmi, run ascend-dmi
                                      -h.
  --modules=<MODULE>[,<MODULE>]...    Specifies the modules whose information is to be
                                      collected. Use commas (,) to separate multiple
                                      modules. The options are ascend and system. By
                                      default, all modules are collected, which is
                                      equivalent to module=all.
                                      [all]
  --ascend-path=<PATH>                Specifies the installation path of the Ascend
                                      software.
                                      [/usr/local/Ascend]
```

3.4.2 查看版本信息

查看ascend-log-collect.sh脚本版本信息。

表 3-2 参数说明

参数	说明	是否必填
[-v, --version]	查看ascend-log-collect.sh脚本版本信息。	是

使用实例

ascend-log-collect.sh --version

```
[HwHiAiUser@localhost ~]$ ascend-log-collect.sh --version
ascend-log-collect.sh
Version: 20.10.0
```

3.4.3 收集日志

日志相关参数说明参见**表3-3**。

表 3-3 参数说明

参数	说明	是否必填
--output-file=<FILENAME>	指定收集日志的输出路径和输出文件的名称，文件的扩展名建议为.tar.gz。不指定“--output-file”参数时，默认在当前路径生成名称为ascend-report-<hostname>-<YYYYMMDDhhmmss>.tar.gz的文件。只指定输出文件的名称，不指定输出路径时，默认在当前路径生成指定名称的文件。	否
--safe	调用ascend-dmi，以level 0模式进行昇腾NPU诊断，此时不会执行功耗测试。如需了解ascend-dmi的更多信息，执行命令**ascend-dmi -h**。 此选项默认开启。	否
--extra	调用ascend-dmi，以level 1模式进行昇腾NPU诊断。此时执行功耗测试，会引起AI core占用率增加和芯片温度升高，收集结束后恢复正常。如需了解ascend-dmi的更多信息，执行命令**ascend-dmi -h**。	否
--modules=<MODULE>[,<MODULE>]…	指定信息收集的模块，多个模块之间用逗号分隔。可选的模块有：ascend和system。默认收集所有模块，等同module=all。	否
--ascend-path=<PATH>	指定昇腾软件安装路径，默认为“/usr/local/Ascend”。	否

使用实例

- ascend-log-collect.sh --safe（以safe模式执行）

```
[HwHiAiUser@localhost ~]$ ascend-log-collect.sh --safe
Collecting system information
Collecting ascend package info
Collecting ascend-dmi results
Collecting npu-smi results
Collecting ascend logs
Done
```


- ascend-log-collect.sh --extra（以extra模式执行）

```
[HwHiAiUser@localhost ~]$ ascend-log-collect.sh --extra
Collecting system information
Collecting ascend package info
Collecting ascend-dmi results
Collecting npu-smi results
Collecting ascend logs
Done
```

- ascend-log-collect.sh --output-file=./result.tar.gz（指定输出文件执行）

```
[HwHiAiUser@localhost ~]$ ascend-log-collect.sh --output-file=./result.tar.gz
Collecting system information
Collecting ascend package info
Collecting ascend-dmi results
Collecting npu-smi results
Collecting ascend logs
Done
```

- (--modules=ascend)：ascend-log-collect.sh --modules=ascend（只收集昇腾NPU健康信息和昇腾软件日志）

```
[HwHiAiUser@localhost ~]$ ascend-log-collect.sh --modules=ascend
Collecting ascend package info
Collecting ascend-dmi results
Collecting npu-smi results
Collecting ascend logs
Done
```


4 接口介绍

4.1 简介

“Ascend-DMI”工具提供的功能接口包含兼容性检测接口、诊断检测接口、硬件检测接口、压测接口、拓扑查询接口。用户可调用接口完成对应的功能测试或者信息查询。

4.2 兼容性检测接口介绍

4.2.1 AdmiCompatibilityCheck 接口原型

函数原型

**int32_t AdmiCompatibilityCheck(const char *path,
AdmiCompatibilitySummary *compatibleRet)**

功能说明

完成软件与硬件之间的兼容性检测。

参数说明

参数名	输入/输出	类型	描述
path	输入	const char*	用户指定的软件包路径，如果不指定，则path为“\0”。
compatibleRet	输出	AdmiCompatibilitySummary*	兼容性检测结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
char path[] = "\0";
AdmiCompatibilitySummary summary = {0};
int32_t ret = AdmiCompatibilityCheck(path, &summary);
if (ret != ADMI_OK) {
    //记录日志
    return ret;
}
......
```

4.3 诊断检测接口介绍

4.3.1 AdmiTestDiagnosis 接口原型

函数原型

int32_t AdmiTestDiagnosis(const AdmiTestDiagnosisOption* option, AdmiTestDiagnosisResult* result)

功能说明

故障诊断。

参数说明

参数名	输入/输出	类型	描述
option	输入	const **AdmiTestDiagnosisOption***	故障诊断选项。
result	输出	**AdmiTestDiagnosisResult***	故障诊断结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiTestDiagnosisOption testDiagnosisParam = {
    -1，// 板卡ID，可通过调用接口AdmiGetProductInfo获取，-1代表对所有板卡进行诊断。
    0    // 诊断级别。
};
AdmiTestDiagnosisResult diagnosisResult;
diagnosisResult.type = DIAGNOSIS_INVALID;
int32_t ret = AdmiTestDiagnosis(&testDiagnosisParam, &diagnosisResult);
......
```

4.4 硬件检测接口介绍

4.4.1 AdmiGetAllDeviceIdInfo 接口原型

函数原型

int32_t AdmiGetAllDeviceIdInfo(AdmiDeviceIdInfo *idInfo)

功能说明

查询所有设备ID。

参数说明

参数名	输入/输出	类型	描述
idInfo	输出	AdmiDeviceIdInfo*	设备ID信息。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiDeviceIdInfo devIdInfo;
int32_t ret = AdmiGetAllDeviceIdInfo(&devIdInfo);
......
```

4.4.2 AdmiGetProductInfo 接口原型

函数原型

int32_t AdmiGetProductInfo(AdmiProductInfo* productInfo)

功能说明

查询产品信息。

参数说明

参数名	输入/输出	类型	描述
productInfo	输出	AdmiProductInfo*	产品信息，包含产品类型、产品名称、板卡数量、板卡信息。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiProductInfo productInfo;
int32_t ret = AdmiGetProductInfo(&productInfo);
......
```

4.4.3 AdmiGetCardInfo 接口原型

函数原型

int32_t AdmiGetCardInfo(int32_t cardId, AdmiCardInfo* cardInfo)

功能说明

指定板卡ID查询板卡信息。

参数说明

参数名	输入/输出	类型	描述
cardId	输入	int32_t	板卡ID。
cardInfo	输出	AdmiCardInfo*	板卡信息。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t cardId = 1;  // 板卡ID，需要根据实际情况指定
AdmiCardInfo cardInfo;
int32_t ret = AdmiGetCardInfo(cardId, &cardInfo);
......
```

4.4.4 AdmiGetCardIdByDeviceId 接口原型

函数原型

int32_t AdmiGetCardIdByDeviceId(int32_t deviceId, int32_t* cardId)

功能说明

通过指定设备ID查询板卡ID。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	设备ID。
cardId	输出	int32_t*	板卡ID。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t deviceId = 1; // 设备ID，需要根据实际情况指定。
int32_t cardId;
int32_t ret = AdmiGetCardIdByDeviceId(deviceId, &cardId);
......
```

4.4.5 AdmiGetDeviceErrorCode 接口原型

函数原型

int32_t AdmiGetDeviceErrorCode(int32_t deviceId, int32_t *count, uint32_t *errorCode)

功能说明

指定设备ID查询错误码。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	设备ID。
count	输出	int32_t*	错误码数量。
errorCode	输出	uint32_t*	错误码列表数组，传入数组长度至少为ADMI_DEV_ERROR_STR_MAX_LEN。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t deviceId = 1;  // 需要根据实际情况指定。
int32_t errorCount;
uint32_t errorCode[ADMI_DEV_ERROR_CODE_MAX_NUM];
int32_t ret = AdmiGetDeviceErrorCode(deviceId, &errorCount, errorCode);
......
```

4.4.6 AdmiGetDeviceErrorString 接口原型

函数原型

int32_t AdmiGetDeviceErrorString(int32_t deviceId, uint32_t errorCode, char *errStr, int32_t size)

功能说明

指定设备ID、错误码获取对应错误信息字符串。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	设备ID。
errorCode	输入	uint32_t	错误码。
errStr	输出	char *	错误信息。
size	输入	int32_t	错误信息数组长度。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t deviceId = 1; // 需要根据实际情况指定。
int32_t errorCode;    // 可通过调用AdmiGetDeviceErrorCode获取。
char errorString[ADMI_DEV_ERROR_STR_MAX_LEN];
AdmiGetDeviceErrorString(devId, errorCode, errorString, ADMI_DEV_ERROR_STR_MAX_LEN);
......
```

4.4.7 AdmiGetDeviceBaseInfo 接口原型

函数原型

int32_t AdmiGetDeviceBaseInfo(int32_t deviceId, AdmiDeviceBaseInfo* deviceBaseInfo)

功能说明

指定设备ID获取设备基本信息。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	设备ID。
deviceBaseInfo	输出	AdmiDeviceBaseInfo*	设备基本信息。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t deviceId = 0;  // 需要根据实际情况指定。
AdmiDeviceBaseInfo deviceBaseInfo;
int32_t ret = AdmiGetDeviceBaseInfo(deviceId, &deviceBaseInfo);
......
```


4.4.8 AdmiGetAllDeviceBaseInfo 接口原型

函数原型

int32_t AdmiGetAllDeviceBaseInfo(uint32_t deviceCount, AdmiDeviceBaseInfo* deviceBaseInfos)

功能说明

根据设备数量获取所有设备基本信息。

参数说明

参数名	输入/输出	类型	描述
deviceCount	输入	uint32_t	设备数量。
deviceBaseInfos	输出	AdmiDeviceBaseInfo*	设备基本信息。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiDeviceIdInfo devIdInfo;
AdmiGetAllDeviceIdInfo(&devIdInfo);
AdmiDeviceBaseInfo *pDevBaseInfo = (AdmiDeviceBaseInfo *)malloc(
    sizeof(AdmiDeviceBaseInfo) * devIdInfo.count);
int32_t ret = AdmiGetAllDeviceBaseInfo(devIdInfo.count, pDevBaseInfo);
......
free(pDevBaseInfo);
```


4.4.9 AdmiGetDeviceDetailInfo 接口原型

函数原型

int32_t AdmiGetDeviceDetailInfo(int32_t deviceId, AdmiDeviceDetailInfo* deviceDetailInfo)

功能说明

指定设备ID获取设备详细信息。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	设备ID。
deviceDetailInfo	输出	AdmiDeviceDetailInfo*	设备详细信息。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t deviceId = 1;  // 需要根据实际情况指定。
AdmiDeviceDetailInfo detailInfo;
int32_t ret = AdmiGetDeviceDetailInfo(deviceId, &detailInfo);
......
```

4.4.10 AdmiGetAllDeviceDetailInfo 接口原型

函数原型

int32_t AdmiGetAllDeviceDetailInfo(uint32_t deviceCount, AdmiDeviceDetailInfo* deviceDetailInfos)

功能说明

查询所有设备详细信息。

参数说明

参数名	输入/输出	类型	描述
deviceCount	输入	uint32_t	设备数量。
deviceDetailInfos	输出	AdmiDeviceDetailInfo*	设备详细信息。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiDeviceIdInfo devIdInfo;
AdmiGetAllDeviceIdInfo(&devIdInfo);
AdmiDeviceDetailInfo *pDevDetailInfo = (AdmiDeviceDetailInfo *)malloc(
    sizeof(AdmiDeviceDetailInfo) * devIdInfo.count);
int32_t ret = AdmiGetAllDeviceDetailInfo(devIdInfo.count, pDevDetailInfo);
......
free(pDevDetailInfo);
```

4.4.11 AdmiGetCardPower 接口原型

函数原型

int32_t AdmiGetCardPower(int32_t cardId, AdmiCardPwrInfo* powerInfo)

功能说明

获取指定板卡功耗信息。

参数说明

参数名	输入/输出	类型	描述
cardId	输入	int32_t	板卡ID。
powerInfo	输出	AdmiCardPwrInfo*	功耗信息。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int32_t cardId = 1;  // 需要根据实际情况指定。
AdmiCardPwrInfo powerInfo;
int32_t ret = AdmiGetCardPower(cardId, &powerInfo);
......
```

4.4.12 AdmiGetDeviceTopoInfo 接口原型

函数原型

int32_t AdmiGetDeviceTopoInfo(AdmiTopoInfo* topoInfo)

功能说明

查询芯片的总线拓扑关系。

参数说明

参数名	输入/输出	类型	描述
topoInfo	输出	AdmiTopoInfo*	总线拓扑信息。

返回值

类型	描述
int32_t	处理结果： ● 返回成功：ADMI_OK（0） ● 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiTopoInfo topoInfo = {0};
int32_t ret;

ret = AdmiGetDeviceTopoInfo(&topoInfo);
if (ret != 0) {
    // 记录日志
    return;
}
......
```

4.5 压测接口介绍

4.5.1 AdmiTestFlops 接口原型

函数原型

int32_t AdmiTestFlops(int32_t deviceId, AdmiTestFlopsOption* option, AdmiComputingCap* caps)

功能说明

测试指定Device ID的算力。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	设备ID，取值范围为[0，63]，当前实际支持的范围通过AdmiGetAllDeviceIdInfo获取。

参数名	输入/输出	类型	描述
option	输入	AdmiTestFlopsOption*	算子测试参数设置。
caps	输出	AdmiComputingCap*	算子输出结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int ret = 0;
int deviceId = 0;
AdmiTestFlopsOption option = {0};
AdmiComputingCap caps = {0};
option.excuteTimes = 10;
ret = AdmiTestFlops(deviceId, &option, &caps);
if(ret != 0) {
    //todo：记录日志
    return ret;
}
......
```

4.5.2 AdmiTestD2DBandwidth 接口原型

函数原型

int32_t AdmiTestD2DBandwidth(int32_t deviceId, const AdmiTestBandwidthOption* option, AdmiD2DBandwidthInfo* testRslt);

功能说明

检测指定Device一侧到另一侧的传输速率和时延。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	指定设备编号，取值范围为[0，63]，当前实际支持的范围通过AdmiGetAllDeviceIdInfo获取。
option	输入	const **AdmiTestBandwidthOption***	D2D带宽测试需要的输入参数，包括测试执行次数，传输数据大小等。
testRslt	输出	**AdmiD2DBandwidthInfo***	D2D带宽测试结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiTestBandwidthOption bandwidthOption = {0};
AdmiD2DBandwidthInfo d2dResult = {0};
int32_t deviceId = 0;
bandwidthOption.dataTransSize = 64*1024*1024;
bandwidthOption.excuteTimes = 5;
bandwidthOption.dataTransMode = ADMI_DATATRANS_MODE_STEP;
int32_t ret = AdmiTestD2DBandwidth(deviceId, &bandwidthOption, &d2dResult);
if (ret != ADMI_OK) {
    //记录日志
    return ret;
}
.....
```


4.5.3 AdmiTestH2DBandwidth 接口原型

函数原型

int32_t AdmiTestH2DBandwidth(int32_t deviceId, const AdmiTestBandwidthOption* option, AdmiH2DBandwidthInfo* testRslt);

功能说明

检测Host侧到指定Device侧的传输速率和时延。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	指定设备编号，取值范围为[0，63]，当前实际支持的范围通过AdmiGetAllDeviceIdInfo获取。
option	输入	const **AdmiTestBandwidthOption***	H2D带宽测试需要的输入参数，包括测试执行次数，传输数据大小等。
testRslt	输出	**AdmiH2DBandwidthInfo***	H2D带宽测试结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiTestBandwidthOption bandwidthOption = {0};
AdmiH2DBandwidthInfo h2dResult = {0};
int32_t deviceId = 0;
bandwidthOption.dataTransSize = 64*1024*1024;
bandwidthOption.excuteTimes = 5;
bandwidthOption.dataTransMode = ADMI_DATATRANS_MODE_STEP;
int32_t ret = AdmiTestH2DBandwidth(deviceId, &bandwidthOption, &d2dResult);
if (ret != ADMI_OK) {
    //记录日志
    return ret;
}
.....
```

4.5.4 AdmiTestD2HBandwidth 接口原型

函数原型

int32_t AdmiTestD2HBandwidth(int32_t deviceId, const AdmiTestBandwidthOption* option, AdmiD2HBandwidthInfo* testRslt);

功能说明

检测指定Device侧到Host侧的传输速率和时延。

参数说明

参数名	输入/输出	类型	描述
deviceId	输入	int32_t	指定设备编号，取值范围为[0，63]，当前实际支持的范围通过AdmiGetAllDeviceIdInfo获取。
option	输入	const **AdmiTestBandwidthOption***	D2H带宽测试需要的输入参数，包括测试执行次数，传输数据大小等。
testRslt	输出	**AdmiD2HBandwidthInfo***	D2H带宽测试结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
AdmiTestBandwidthOption bandwidthOption = {0};
AdmiD2HBandwidthInfo d2hResult = {0};
int32_t deviceId = 0;
bandwidthOption.dataTransSize = 64*1024*1024;
bandwidthOption.excuteTimes = 5;
bandwidthOption.dataTransMode = ADMI_DATATRANS_MODE_STEP;
int32_t ret = AdmiTestD2HBandwidth(deviceId, &bandwidthOption, &d2dResult);
if (ret != ADMI_OK) {
    //记录日志
    return ret;
}
.....
```

4.5.5 AdmiTestP2PBandwidth 接口原型

函数原型

int32_t AdmiTestP2PBandwidth(int32_t srcDeviceId, int32_t dstDeviceId, const AdmiTestBandwidthOption* option, AdmiP2PBandwidthInfo* testRslt);

功能说明

测试指定Device到另一Device的传输速率和时延。

参数说明

参数名	输入/输出	类型	描述
srcDeviceId	输入	int32_t	指定源的设备编号，取值范围为[0，63]，当前实际支持的范围通过AdmiGetAllDeviceIdInfo获取。
dstDeviceId	输入	int32_t	指定目的的设备编号，取值范围为[0，63]，当前实际支持的范围通过AdmiGetAllDeviceIdInfo获取。

参数名	输入/输出	类型	描述
option	输入	const AdmiTestBandwidthOption*	P2P带宽测试需要的输入参数，包括测试执行次数，传输数据大小等。
testRslt	输出	AdmiP2PBandwidthInfo*	P2P带宽测试结果。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

仅支持Ascend 910系列芯片设备。

源的设备编号与目的的设备编号不能一样，否则报错。

调用示例

```
AdmiTestBandwidthOption bandwidthOption = {0};
AdmiP2PBandwidthInfo p2pResult = {0};
int32_t srcDeviceId = 0;
int32_t dstDeviceId = 1;
bandwidthOption.dataTransSize = 64*1024*1024;
bandwidthOption.excuteTimes = 5;
bandwidthOption.dataTransMode = ADMI_DATATRANS_MODE_STEP;
int32_t ret = AdmiTestP2PBandwidth(srcDeviceId, dstDeviceId, &bandwidthOption, &d2dResult);
if (ret != ADMI_OK) {
    //记录日志
    return ret;
}
.....
```

4.5.6 AdmiTestPower 接口原型

函数原型

int32_t AdmiTestPower(int32_t cardId, const AdmiTestPowerOption* option, AdmiTestPowerCallback* callback)

功能说明

在AI Core利用率100%的情况下，测试Atlas板卡的功耗值。

参数说明

参数名	输入/输出	类型	描述
cardId	输入	int32_t	Atlas 板卡ID，值为-1表示测试所有Atlas板卡的功耗值。Atlas 板卡的编号通过AdmiGetProductInfo获取。
option	输入	const **AdmiTestPowerOption***	功耗测试参数设置。
callback	输入	**AdmiTestPowerCallback***	功耗输出回调函数。

返回值

类型	描述
int32_t	处理结果： • 返回成功：ADMI_OK（0） • 失败返回错误码

异常处理

无。

约束说明

无。

调用示例

```
int ret = 0;
int cardId = -1;
AdmiTestPowerOption option = {0};
AdmiTestPowerCallback callback = {0};
option.duration = 60;
option.powerInfoReportCycle = 5;
callback.powerInfoReportCallback = PowerInfoOutputProc;
callback.temperatureAlarmCallback = TemperatureAlarmProc;
ret = AdmiTestPower(cardId, &option, &callback);
if(ret != 0) {
    //todo：记录日志
    return ret;
}
void PowerInfoOutputProc(AdmiPowerDetailInfo* powerInfo)
{
   if (powerInfo != nullptr) {
      // todo: 输出功耗信息
```



```
  }
}
void TemperatureAlarmProc(AdmiTemperatureAlarmInfo* alarmInfo)
{
  if (alarmInfo != nullptr) {
    // todo: 输出温度信息
  }
}
......
```

4.6 调用示例

各接口介绍章节中的调用示例为各接口的调用片段，本章节调用示例以调用AdmiTestD2DBandwidth接口为例，给出完整调用过程并介绍所需加入的头文件（如“admi_interface.h”头文件）、调用接口的逻辑等。

用户可在环境中toolbox工具安装目录的“Ascend-DMI/include/”目录下获取“admi_interface.h”文件，所在目录参考为“/usr/local/Ascend/toolbox/latest/Ascend-DMI/include/admi_interface.h”。

示例代码文件 AdmiTestD2DBandwidth.c

```
#include <stdio.h>
#include <stdlib.h>
#include "admi_interface.h"

int main(int argc, char ** argv)
{
    int32_t ret;
    AdmiTestBandwidthOption bandwidthOption = {0};
    AdmiD2DBandwidthInfo d2dResult = {0};
    int32_t deviceId = 0;
    bandwidthOption.dataTransSize = 64*1024*1024;
    bandwidthOption.excuteTimes = 5;
    bandwidthOption.dataTransMode = ADMI_DATATRANS_MODE_STEP;

    int32_t ret = AdmiTestD2DBandwidth(deviceId, &bandwidthOption, &d2dResult);
    if (ret != 0) {
        printf("call AdmiTestD2DBandwidth error, ret = %d.", ret);
        return ret;
    }

    if (deviceCount != 1) {
        return 0;
    }
    printf("device info: deviceid is %d, devicetype is %d.\n", d2dResult.d2dBandwidthInfo[0].deviceId,
d2dResult.d2dBandwidthInfo[0].deviceType);
    for (int32_t loop = 0; loop < d2dResult.d2dBandwidthInfo[0].rateInfoCount;loop++) {
        printf("size: %d, rate: %2f, latency: %2f.\n", d2dResult.d2dBandwidthInfo[0].rateInfo[loop].dataSize,
d2dResult.d2dBandwidthInfo[0].rateInfo[loop].rate,
d2dResult.d2dBandwidthInfo[0].rateInfo[loop].timeDelay);
    }
    return ret;
}
```

使用调用示例

步骤1 以**HwHiAiUser**用户将AdmiTestD2DBandwidth.c、admi_interface.h传到Host侧服务器的同一个目录下。

步骤2 以**HwHiAiUser**用户登录到Host侧服务器。

步骤3 执行如下命令，编译AdmiTestD2DBandwidth.c中的代码，生成可执行文件AdmiTestD2DBandwidth。编译时需要使用驱动中的dsmi、dcmi、acl相关的驱动如命令所示。

```
gcc AdmiTestD2DBandwidth.c libascend_dmi.so libdcmi.so  libdrvdsmi_host.so libdrvhdc_host.so libascendcl.so libc_sec.so -L -o bandwidth_test
```

步骤4 执行可执行文件bandwidth_test。

```
./bandwidth_test
```

----结束

4.7 附录

4.7.1 结构体总结

AdmiTestFlopsOption

```
typedef struct {
    int32_t excuteTimes;                    /* 测试执行次数，单位：100万次 */
} AdmiTestFlopsOption;
```

AdmiComputingCap

```
typedef struct {
    int32_t deviceId;                       /* 昇腾处理器的Device ID */
    int32_t excuteTimes;                    /* 测试执行的次数，单位：100万次 */
    uint32_t duration;                      /* 测试执行的时间，单位：ms */
    float flops;                            /* 昇腾处理器的计算能力，单位：TFLOPS */
} AdmiComputingCap;
```

AdmiTestBandwidthOption

```
typedef struct {
    uint32_t excuteTimes;       /* 测试执行次数 */
    uint32_t dataTransMode;     /* 表示测试传输数据模式:当取值为ADMI_DATATRANS_MODE_LADDER时,表示传输数据从2K到16M遍历 */
    uint32_t dataTransSize;     /* 表示传输数据的固定大小,取值范围[1, 512M]，单位: Byte，仅当dataTransMode等于ADMI_DATATRANS_MODE_STEP有效。 */
} AdmiTestBandwidthOption;
```

AdmiBandwidthRateInfo

```
typedef struct {
    uint32_t dataSize;                      /* 测试时传输数据大小 */
    double rate;                            /* 传输速率，单位: Mbps */
    double timeDelay;                       /* 传输时间，单位: us */
} AdmiBandwidthRateInfo;
```

AdmiBandwidthInfo

```
typedef struct {
    int32_t deviceId;                       /* 昇腾处理器的设备编号 */
    AdmiDeviceType deviceType;              /* 昇腾处理器设备类型 */
    uint32_t excuteTimes;                   /* 测试执行的次数 */
    uint32_t rateInfoCount;                 /* 速率信息个数 */
    AdmiBandwidthRateInfo rateInfo[ADMI_MAX_LADDER_NUM];       /* 速率信息列表 */
} AdmiBandwidthInfo;
```

AdmiH2DBandwidthInfo

```
typedef struct {
    uint32_t deviceCount;                   /* 昇腾处理器的设备的数量 */
```



```
    AdmiBandwidthInfo h2dBandwidthInfo[ADMI_MAX_DEVICE_COUNT];  /* H2D带宽测试结果信息列表 */
}AdmiH2DBandwidthInfo;
```

AdmiD2HBandwidthInfo

```
typedef struct {
    uint32_t deviceCount;                                   /* 昇腾处理器的设备的数量 */
    AdmiBandwidthInfo d2hBandwidthInfo[ADMI_MAX_DEVICE_COUNT];  /* D2H带宽测试结果信息列表 */
} AdmiD2HBandwidthInfo;
```

AdmiD2DBandwidthInfo

```
typedef struct {
    uint32_t deviceCount;                                   /* 昇腾处理器的设备的数量 */
    AdmiBandwidthInfo d2dBandwidthInfo[ADMI_MAX_DEVICE_COUNT];  /* D2D带宽测试结果信息列表 */
} AdmiD2DBandwidthInfo;
```

AdmiSrcP2PBandwidthInfo

```
typedef struct {
    int32_t srcDeviceId;                                  /* 昇腾处理器的设备的数量 */
    AdmiDeviceType srcDeviceType;                             /* 昇腾处理器设备类型 */
    uint32_t dstDeviceCount;                                /* 作为目标的昇腾处理器设备个数 */
    AdmiBandwidthInfo dstBandwidthInfo[ADMI_MAX_DEVICE_COUNT];  /* 目标侧的P2P带宽单侧测试结果信
息列表 */
    uint32_t dstBiDeviceCount;                                /* 作为目标的昇腾处理器设备个数 */
    AdmiBandwidthInfo dstBiBandwidthInfo[ADMI_MAX_DEVICE_COUNT];/* 目标侧的P2P带宽双侧测试结果
信息列表 */
} AdmiSrcP2PBandwidthInfo;
```

AdmiP2PBandwidthInfo

```
typedef struct {
    uint32_t srcDeviceCount;                                    /* 作为源的昇腾处理器设备的个数 */
    AdmiSrcP2PBandwidthInfo srcBandwidthInfo[ADMI_MAX_DEVICE_COUNT];      /* 源侧的P2P带宽测试结
果信息列表 */
} AdmiP2PBandwidthInfo;
```

AdmiDeviceIdInfo

```
typedef struct {
    int32_t deviceCount;                        /* device 总数 */
    int32_t deviceId[ADMI_MAX_DEVICE_COUNT];        /* device id */
} AdmiDeviceIdInfo;
```

AdmiCardInfo

```
typedef struct  {
    int32_t cardType;                       /* 卡类型 */
    char cardName[ADMI_MAX_CARD_NAME_LEN];          /* 卡名称 */
    char manufactuer[ADMI_MAX_CARD_ELABLE_LEN];     /* 厂商 */
    char serialNum[ADMI_MAX_CARD_ELABLE_LEN];       /* 卡SN号 */
    int32_t cardId;                         /* 卡号 */
    uint32_t deviceCount;                       /* 当前卡中的device个数 */
    int32_t deviceId[ADMI_MAX_DEVICE_NUM_IN_CARD];  /* 当前卡上的device id */
} AdmiCardInfo;
```

AdmiProductInfo

```
typedef struct {
    int32_t productType;                        /* 产品类型 */
    char productName[ADMI_MAX_PRODUCT_NAME_LEN];    /* 产品名称 */
    int32_t cardCount;
    AdmiCardInfo cardInfo[ADMI_MAX_CARD_NUM];
} AdmiProductInfo;
```


AdmiCardPwrInfo

```
typedef struct {
    float curPwr;
    float ratePwr;
} AdmiCardPwrInfo;
```

PcieId

```
typedef struct {
    uint32_t domain;
    uint32_t bus;
    uint32_t dev;
    uint32_t func;
} PcieId;
```

PcieCfgInfo

```
typedef struct {
    uint32_t deviceId;
    uint32_t venderId;
    uint32_t subVenderId;
    uint32_t subDeviceId;
    uint32_t devClass;
    uint32_t primaryBus;
    uint32_t secondaryBus;
    uint32_t capSpeed;
    uint32_t capWidth;
    uint32_t staSpeed;
    uint32_t staWidth;
} PcieCfgInfo;
```

AdmiDeviceBaseInfo

```
typedef struct {
    int32_t deviceId;                       /* 设备号 */
    AdmiDeviceType deviceType;              /* 设备类型 */
    char deviceName[ADMI_MAX_DEVICE_NAME_LEN];  /* 设备名称 */
    int32_t cardId;                         /* 板卡ID号 */
    int32_t chipId;                         /* 芯片ID号 */
    uint64_t memorySize;                    /* 内存容量,单位MB */
    uint32_t memoryUsage;                   /* 内存使用率 */
    float ratedPower;                       /* 额定功耗 */
    float currentPower;                     /* 当前功耗 */
    uint32_t aiCoreUsage;                   /* AI Core 使用率, 单位% */
    float temperature;                      /* NPU 温度 */
    float voltage;                          /* 设备电压 */
    uint32_t health;                        /* 健康度 */
    char busId[ADMI_MAX_BUS_ID_LEN];        /* 总线Id */
} AdmiDeviceBaseInfo;
```

AdmiEccDetailInfo

```
typedef struct {
    int32_t eccEnable;
    uint32_t singleBitErrCnt;
    uint32_t doubleBitErrCnt;
} AdmiEccDetailInfo;
```

AdmiEccInfo

```
typedef struct {
    AdmiEccType eccType;
    AdmiEccDetailInfo eccInfo;
} AdmiEccInfo;
```


AdmiDeviceDetailInfo

```
typedef struct {
    AdmiDeviceBaseInfo deviceBaseInfo;              /* 设备基础信息 */

    int32_t aiCoreCount;                            /* AI Core 数量 */
    int32_t cpuCount;                               /* CPU数量 */
    int32_t aiCpuCount;                             /* AI CPU 数量 */
    uint32_t aiCpuUsage;                            /* AI CPU 使用率, 单位% */
    int32_t ctrlCpuCount;                           /* 控制CPU 数量 */
    uint32_t ctrlCpuFreq;                           /* 控制CPU 频率 */
    uint32_t ctrlCpuUsage;                          /* 控制CPU 使用率, 单位% */
    int32_t cubeCount;                              /* Cube 数量 */
    int32_t vectorCount;                            /* Vector数量 */
    uint32_t frequency;                             /* 内存频率, 单位MHz */
    uint32_t memoryBusUsage;                        /* 内存带宽使用率 */
    AdmiEccInfo eccAarry[ADMI_MAX_ECC_TYPE_NUM];    /* ECC 信息 */
    PcieId pcieId;                                  /* PCIE ID */
    PcieCfgInfo pcieCfg;                            /* PCIE Cfg */
    char pcieAffinityCpu[ADMI_AFFINITY_CPU_MAX_LEN];  /* 亲和CPU列表 */
    char pcieCapSpeed[ADMI_PCIE_SPEED_STR_MAX_LEN];   /* PCIE link speed */
    char pcieStaSpeed[ADMI_PCIE_SPEED_STR_MAX_LEN];   /* PCIE link speed */
    uint32_t socDie[ADMI_MAX_DEVICE_DIE_ID_LEN];      /* device DIE ID */
} AdmiDeviceDetailInfo;
```

AdmiPowerDeviceInfo

```
typedef struct {
    char deviceName[ADMI_MAX_DEVICE_NAME_LEN];          /* 昇腾处理器的名称 */
    int32_t deviceId;                                   /* 昇腾处理器的Device ID */
    int32_t chipId;                                     /* 昇腾处理器的芯片ID，如果有4个芯片，则编号为0、1、2、3。
*/
    uint32_t health;                                    /* 昇腾处理器的健康状态 */
    uint32_t aiCoreUsage;                               /* 昇腾处理器的AI Core利用率 */
    float temperature;                                  /* 昇腾处理器的温度 */
    float voltage;                                      /* 昇腾处理器的电压 */
} AdmiPowerDeviceInfo;
```

AdmiCardPowerInfo

```
typedef struct {
    char cardName[ADMI_MAX_CARD_NAME_LEN];              /* Atlas板卡的类型 */
    int32_t cardId;                                     /* Atlas板卡的编号 */
    uint32_t deviceCount;                               /* 昇腾处理器的个数 */
    float power;                                        /* Atlas板卡的功耗值 */
    int32_t averageTemp;                                /* Atlas板卡中昇腾处理器的平均温度 */
    int32_t averageVoltage;                             /* Atlas板卡中昇腾处理器的平均电压 */
    uint32_t averageAiCoreUsage;                        /* Atlas板卡中昇腾处理器的平均AI Core利用率 */
    AdmiPowerDeviceInfo deviceInfo[ADMI_MAX_DEVICE_COUNT];  /* 昇腾处理器的信息 */
} AdmiCardPowerInfo;
```

AdmiPowerDetailInfo

```
typedef struct {
    int32_t cardCount;                                  /* Atlas板卡的个数 */
    AdmiCardPowerInfo cardPowerInfo[ADMI_MAX_CARD_NUM];     /* Atlas板卡的功耗信息 */
} AdmiPowerDetailInfo;
```

AdmiTestPowerOption

```
typedef struct {
    int32_t duration;                                   /* 测试执行的时间，单位：s */
    int32_t powerInfoReportCycle;                       /* 功耗值循环输出的时间，单位：s */
} AdmiTestPowerOption;
```


AdmiCardPowerSummaryInfo

```
typedef struct {
    int32_t cardId;
    float maxPower;
    float averagePower;
    int32_t maxAiCore;
    int32_t averageAiCore;
    int32_t maxTemp;
    int32_t averageTemp;
    int32_t maxVoltage;
    int32_t averageVoltage;
} AdmiCardPowerSummaryInfo;
```

AdmiTestPowerResult

```
typedef struct {
    int32_t duration;
    int32_t reportCycle;
    int32_t cardCount;
    AdmiCardPowerSummaryInfo cardPowerSummaryInfo[ADMI_MAX_CARD_NUM];
} AdmiTestPowerResult;
```

AdmiTemperatureAlarmInfo

```
typedef struct {
    int32_t deviceId;           /* 昇腾处理器的Device ID */
    float currentTemperature;    /* 昇腾处理器的温度 */
} AdmiTemperatureAlarmInfo;
```

AdmiTestPowerCallback

```
typedef void (*AdmiPowerInfoReportCallback)(AdmiPowerDetailInfo *);
typedef void (*AdmiTemperatureAlarmCallback)(AdmiTemperatureAlarmInfo *);
typedef struct {
    AdmiPowerInfoReportCallback powerInfoReportCallback;       /* 功耗值输出回调函数 */
    AdmiTemperatureAlarmCallback temperatureAlarmCallback;     /* 温度告警回调函数 */
} AdmiTestPowerCallback;
```

AdmiTopoInfo

```
typedef struct {
    uint32_t deviceCount;                                   /* 设备个数 */
    int32_t deviceIdList[ADMI_MAX_DEVICE_COUNT];              /* 设备编号列表 */
    char affinityCpuInfo[ADMI_MAX_DEVICE_COUNT][ADMI_AFFINITY_CPU_MAX_LEN];
    /* 设备的亲和CPU列表，与deviceIdList的数组下标一一对应 */
    AdmiTopoType topoTypeMatrix[ADMI_MAX_DEVICE_COUNT][ADMI_MAX_DEVICE_COUNT];
    /* 设备两两之间的拓扑关系，行和列的下标与deviceIdList的数组下标一一对应 */
} AdmiTopoInfo;
```

AdmiTestDiagnosisOption

```
typedef struct {
    int32_t cardId;          /* 板卡的ID */
    int32_t level;           /* 诊断级别 */
} AdmiTestDiagnosisOption;
```

AdmiDiagnosisDeviceInfo

```
typedef struct {
    int32_t deviceId;
    uint32_t health;
    AdmiComputingCap computingCap;
    AdmiBandwidthInfo d2dBandwidthInfo;
    AdmiBandwidthInfo h2dBandwidthInfo;
    AdmiBandwidthInfo d2hBandwidthInfo;
} AdmiDiagnosisDeviceInfo;
```


AdmiDiagnosisCardInfo

```
typedef struct {
    int32_t cardId;
    int32_t numOfDevices;
    AdmiDiagnosisDeviceInfo deviceInfoList[ADMI_MAX_DEVICE_NUM_IN_CARD];
    AdmiCardPowerSummaryInfo cardPowerInfo;
} AdmiDiagnosisCardInfo;
```

AdmiTestDiagnosisResult

```
typedef struct {
    AdmiDiagnosisType type;
    int32_t numOfCards;
    AdmiDiagnosisCardInfo cardList[ADMI_MAX_CARD_NUM];
} AdmiTestDiagnosisResult;
```

AdmiSystemInfo

```
typedef struct {
    char osVersion[ADMI_OS_VERSION_LEN];
    char arch[ADMI_ARCH_LEN];
    char productName[ADMI_MAX_PRODUCT_NAME_LEN];
} AdmiSystemInfo;
```

AdmiPackageInfo

```
typedef struct {
    char packageName[ADMI_MAX_PACKAGE_NAME_LEN];
    char fullpackageName[ADMI_MAX_PACKAGE_NAME_LEN];
    char packageVersion[ADMI_PACKAGE_VERSION_LEN];
    char conditionDescription[ADMI_MAX_CONDITION_DESCRIPTION_LEN];
    AdmiPackageState packageState;
} AdmiPackageInfo;
```

AdmiCompatibilitySummary

```
typedef struct {
    AdmiSystemInfo systemInfo;
    AdmiPackageInfo driverInfo;
    AdmiPackageInfo firmwareInfo;
    int32_t softwareCount;
    AdmiPackageInfo softwareInfo[ADMI_MAX_PACKAGE_NUM];
    int32_t unrecognizedCount;
    AdmiPackageInfo unrecognizedPackages[ADMI_MAX_PACKAGE_NUM];
    int32_t checkResult;
    int32_t upgradeRecommended;
} AdmiCompatibilitySummary;
```

4.7.2 枚举总结

AdmiDeviceType

```
typedef enum {
    ASCEND_310 = 0,              /* Ascend 310 */
    ASCEND_910A,                 /* Ascend 910 A*/
    ASCEND_910B,                 /* Ascend 910 B*/
    ASCEND_910PROA,              /* Ascend 910 Pro A*/
    ASCEND_DEVICE_BUTT
} AdmiDeviceType;
```

AdmiCardType

```
typedef enum  {
    ATLAS_200_3000 = 0,
```



```
    ATLAS_300_3010,
    ATLAS_300_3000,
    ATLAS_300_6000,
    ATLAS_300_6010,
    ATLAS_300_9000_910A,
    ATLAS_300_9000_910B,
    ATLAS_800_9000_910A,
    ATLAS_800_9000_910B,
    ATLAS_800_9000_910PROA,
    ATLAS_800_9010_910A,
    ATLAS_800_9010_910B,
    ATLAS_PRODUCT_BUTT
} AdmiCardType;
```

AdmiEccType

```
typedef enum {
    ADMI_ECC_TYPE_DDR,
    ADMI_ECC_TYPE_SRAM,
    ADMI_ECC_TYPE_HBM,
    ADMI_ECC_TYPE_NPU,
    ADMI_ECC_TYPE_NONE = 0xff
} AdmiEccType;
```

AdmiDataTransMode

```
typedef enum {
    ADMI_DATATRANS_MODE_STEP,
    ADMI_DATATRANS_MODE_LADDER,
    ADMI_DATATRANS_MODE_BUFF
} AdmiDataTransMode;
```

AdmiDiagnosisType

```
typedef enum {
    DIAGNOSIS_LEVEL_0 = 0,
    DIAGNOSIS_LEVEL_1,
    DIAGNOSIS_INVALID,
} AdmiDiagnosisType;
```

AdmiTopoType

```
typedef enum {
    TOPO_TYPE_SELF = 0,     /* 芯片本身 */
    TOPO_TYPE_SYS,          /* 通过PCIe连接且穿过NUMA nodes， nodes之间通过SMP连接，如：QPI、 UPI和
HCCS */
    TOPO_TYPE_NODE,         /* 通过PCIe连接且穿过PCIe host bridge，但是在同个NUMA node */
    TOPO_TYPE_PHB,          /* 通过PCIe连接且穿过同一个CPU的 PCIe host bridge */
    TOPO_TYPE_PXB,          /* 通过PCIe连接且穿过多个PCIe switch */
    TOPO_TYPE_PIX,          /* 通过PCIe连接且穿过同一个PCIe switch */
    TOPO_TYPE_BUTT,         /* 未知关系 */
} AdmiTopoType;
```

AdmiPackageState

```
typedef enum {
    PKG_STAT_OK,                    /* 兼容性检查通过 */
    PKG_STAT_OS_INCOMPATIBLE,       /* 与操作系统版本不兼容 */
    PKG_STAT_ARCH_INCOMPATIBLE,     /* 与系统架构不兼容 */
    PKG_STAT_PRODUCT_INCOMPATIBLE,  /* 与产品形态不兼容 */
    PKG_STAT_PACKAGE_INCOMPATIBLE,  /* 其他软件包版本不满足依赖 */
    PKG_STAT_NOT_INSTALLED,         /* 未安装 */
    PKG_STAT_DEPENDENCIES_UNKNOWN,  /* 依赖未知 */
    PKG_STAT_PACKAGE_UNRECOGNIZED,  /* 包名不识别 */
    PKG_STAT_UNKNOWN                /* 未知状态 */
} AdmiPackageState;
```


4.7.3 宏定义总结

```
#define ADMI_MAX_PRODUCT_NAME_LEN 32                /* 最大昇腾产品名称长度 */
#define ADMI_MAX_CARD_NAME_LEN    32                /* 最大昇腾卡名称长度 */
#define ADMI_MAX_DEVICE_NAME_LEN  32                /* 最大昇腾处理器名称长度 */
#define ADMI_MAX_DEVICE_COUNT 64                    /* 最大昇腾处理器个数 */
#define ADMI_MAX_CARD_NUM 16                        /* 最大卡数 */
#define ADMI_MAX_LADDER_NUM 32                      /* 最大阶梯数 */
#define ADMI_MAX_DEVICE_NUM_IN_CARD 8               /* 单卡上最大NPU芯片数 */
#define ADMI_MAX_BW_FIXED_SIZE 512 * 1024 * 1024    /* 带宽工具支持指定的最大数据传输量 */
#define ADMI_MAX_BW_EXECUTE_TIMES 1000              /* 带宽工具支持指定的最大迭代次数 */
#define ADMI_POWER_DURATION_MIN 60                  /* 功耗工具最小运行时间1分钟，单位：秒，1分钟 */
#define ADMI_POWER_DURATION_MAX 604800              /* 功耗工具最大运行时间7 * 24小时，单位：秒 */
#define ADMI_POWER_INTERVAL_MIN 1                   /* 功耗工具功耗信息最小输出间隔，单位：秒 */
#define ADMI_POWER_INTERVAL_MAX 5                   /* 功耗工具功耗信息最大输出间隔，单位：秒  */

#define ADMI_AFFINITY_CPU_MAX_LEN 128               /* 亲和CPU最大长度，是从系统文件中读取的字符串，
格式：0-15,32-47 */
#define ADMI_PCIE_SPEED_STR_MAX_LEN 16              /* PCIE 链路速度信息字符串长度 */

#define ADMI_MAX_BUS_ID_LEN             16
#define ADMI_MAX_CARD_ELABLE_LEN        256
#define ADMI_MAX_ECC_TYPE_NUM           4
#define ADMI_MAX_DEVICE_DIE_ID_LEN      5

#define ADMI_MAX_PACKAGE_NAME_LEN  64
#define ADMI_MAX_CONDITION_DESCRIPTION_LEN 128
#define ADMI_OS_VERSION_LEN 64
#define ADMI_ARCH_LEN 16
#define ADMI_MAX_PACKAGE_NUM  32
#define ADMI_PACKAGE_VERSION_LEN 30
```


5 参考信息

- 可执行文件路径
  “ascend-dmi”随toolbox安装，所在目录参考为“/usr/local/Ascend/toolbox/latest/Ascend-DMI/bin”。
- 日志路径
  root用户日志路径为“/var/log/ascend-dmi”。
  非root用户日志路径为“${HOME}/var/log/ascend-dmi”。

6 FAQ

6.1 非 root 用户运行 ascend-dmi 命令报错

问题现象

执行ascend-dmi报如下错：

```
[pizi@euler sbin]$ ascend-dmi -i
Error code [0x3] is displayed. Rectify the error based on the log information.
Log file path: /home/pizi/var/log/ascend-dmi/ascend-dmi.log
```

可能原因

dmidecode权限不足

解决方案

在root用户下运行**chmod +s /usr/sbin/dmidecode**和**chmod +s usr/local/sbin/npu-smi**，添加s权限；

然后在非root用户里添加**export PATH=$PATH:/usr/sbin/**。

```
[root@euler sbin]# chmod +s /usr/sbin/dmidecode
[root@euler sbin]#
[root@euler sbin]#
[root@euler sbin]#
[root@euler sbin]# chmod +s /usr/local/sbin/npu-smi
[root@euler sbin]#
```



```
[pizi@euler sbin]$ export PATH=$PATH:/usr/sbin/
[pizi@euler sbin]$ ascend-dmi info
|======================+====================================+=====================|
| ascend-dmi      20.1.0.RC1                       Brief Information                |
|======================+====================================+=====================|
| Card   Type          | NPU Count                          | Real-time Card Power |
+----------------------+------------------------------------+---------------------+
| Chip   Name          | Health           Used Memory       | Temperature   Voltage |
| Device ID            | Bus ID           AI Core Usage     |                     |
|======================+====================================+=====================|
| 0      Atlas300-3010 | 4                                  | NA                  |
+----------------------+------------------------------------+---------------------+
|    0   Ascend310     | OK               2703MB/8192MB     | 51C           0.80V  |
|    0                 | 0000:00:04.0     0%                |                     |
+----------------------+------------------------------------+---------------------+
|    0   Ascend310     | OK               2703MB/8192MB     | 51C           0.80V  |
|    0                 | 0000:00:04.0     0%                |                     |
+----------------------+------------------------------------+---------------------+
|    0   Ascend310     | OK               2703MB/8192MB     | 51C           0.80V  |
|    0                 | 0000:00:04.0     0%                |                     |
+----------------------+------------------------------------+---------------------+
|    0   Ascend310     | OK               2703MB/8192MB     | 51C           0.80V  |
|    0                 | 0000:00:04.0     0%                |                     |
|======================+====================================+=====================|

Warning(s):
 - Get card power failed! Note: card power can only be acquired on host.
[pizi@euler sbin]$ ascend-dmi -bw

Device to Device Test
Device  0: Ascend 310.
-----------------------------------------------------------
 Size(Bytes)  Execute Times  Bandwidth(MB/s)   Latency(us)
-----------------------------------------------------------
         2           5            0.02            78.00
         4           5            0.05            81.66
         8           5            0.10            80.07
        16           5            0.20            77.92
        32           5            0.38            81.02
        64           5            0.75            81.46
       128           5            1.47            82.93
       256           5            3.04            80.47
       512           5            6.10            80.15
      1024           5           12.40            78.80
      2048           5           24.63            79.31
      4096           5           49.02            79.75
      8192           5           93.28            84.04
     16384           5          193.75            80.66
     32768           5          373.56            83.72
     65536           5          707.89            88.33
    131072           5         1318.21            94.93
    262144           5         2454.94           101.96
    524288           5         4840.75           103.35
   1048576           5         8131.30           122.98
   2097152           5        12093.09           165.54
   4194304           5        16204.44           247.44
   8388608           5        14954.73           575.34
```

A 修订记录

发布日期	修改说明
2020-10-15	第一次正式发布。

