Atlas Intelligent Edge Solution-Atlas 500
V100R020C00

Development Environment Installation Guide

Issue 01

Date 2020-09-21

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions

and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

About This Document

Purpose

This document provides guides to get the information about how to install the development environment at Atlas 500, and also give how to build and run a sample project on it.

Intended Audience

This document is intended for engineers who:

- Wants to develop an application for Atlas 500
- Installation and commissioning engineers
- Technical support engineers

Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
DANGER	Indicates a hazard with a high level of risk which, if not avoided, will result in death or serious injury.
WARNING	Indicates a hazard with a medium level of risk which, if not avoided, could result in death or serious injury.
CAUTION	Indicates a hazard with a low level of risk which, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a potentially hazardous situation which, if not avoided, could result in equipment damage, data loss, performance deterioration, or unanticipated results. NOTICE is used to address practices not related to personal injury.
NOTE	Supplements the important information in the main text. NOTE is used to address information not related to personal injury, equipment damage, and environment deterioration.

Change History

Issue	Date	Author	Description
01	2020-09-21	Ascend FAE TEAM	This issue is the first official release. Release for Atlas 500 version 20.0.0.RC1

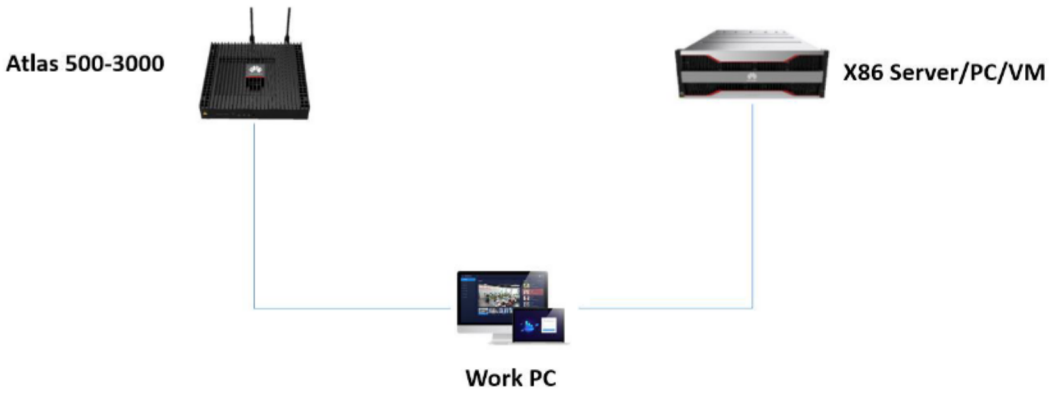
Contents

1 Before you Start

1.1 Hardware Requirements

Step 1 Prepare an x86 server (or PC or Virtual Machine of server) for setting up the development environment.

Step 2 Prepare Atlas 500 for application operating environment.



1.2 Sofrware Requirements

1.2.1 Install OS and Create the Installation User

Install the Ubuntu x86 or Cent OS X86 on the server.

Parameter	Version	How to Obtain

OS	18.04(Ubuntu) 7.6(Cent OS)	Visit the official Ubuntu/CentOS website to download the recommended version: **Ubuntu:** http://old-releases.ubuntu.com/releases/ CentOS: http://vault.centos.org/7.6.1810/isos/x86_64/

1.2.2 Checking the Umask of the root User

Step 1 Log in to the installation environment as the **root** user.

Step 2 Check the **umask** value of the **root** user.

```
umask
```

Step 3 If the umask value is not 0022, append "umask 0022" to the file and command "wq!" to save.

```
vi ~/.bashrc    //open the .bashrc file
source ~/.bashrc    //reload the file
```

1.2.3 Creating Installation and Running Users

Step 1 Create the **HwHiAiUser** user group.

Switch to the **root** user and run the following command to create the **HwHiAiUser** user group:

```
groupadd HwHiAiUser   //Create the HwHiAiUser user group.
```

Step 2 Create the HwHiAiUser user.

```
#Creat your count, the name can be customized
useradd -g HwHiAiUser -d /home/username username
#set the password
passwd username
```

NOTE

- The created running user cannot belong to the **root** user group.
- After the **HwHiAiUser** user is created, do not disable the login authentication function of the user.
- The password validity period is 90 days. You can change the validity period in the **/etc/login.defs** file or using the **chage** command.
- If you want to use the non-root use as installation user,

Step 3 Configuring the Permission of the Installation User (Optional)

Skip the following part if you install as the **root** user.

Before installing the development kit, you need to download the dependencies, which require **sudo apt-get** permission. Run the following commands as the **root** user:

1. Open the **/etc/sudoers** file:

```
chmod u+w /etc/sudoers
vi /etc/sudoers
```

2. Add the following content below **# User privilege specification** of the file:

```
username ALL=(ALL:ALL) NOPASSWD:SETENV:/usr/bin/apt-get, /usr/bin/pip, /bin/tar,
/bin/mkdir, /bin/rm, /bin/sh, /bin/cp, /bin/bash, /usr/bin/make install, /bin/ln
-s /usr/local/python3.7.5/bin/python3 /usr/bin/python3.7, /bin/ln -s
/usr/local/python3.7.5/bin/pip3 /usr/bin/pip3.7, /bin/ln -s
/usr/local/python3.7.5/bin/python3 /usr/bin/python3.7.5, /bin/ln -s
/usr/local/python3.7.5/bin/pip3 /usr/bin/pip3.7.5, /usr/bin/unzip
```

Replace **username** with the name of user who executes the installation.

Replace **apt-get** with **yum** if your OS is Cent OS.

NOTE

Ensure that the last line of the **/etc/sudoers** file is **#includedir /etc/sudoers.d**. Otherwise, add it manually.

3. Run **:wq!** to save the file.

1.2.4 Checking the Source Validity

Development kit installation requires the download of related dependencies. Ensure that the installation environment can be connected to the network.

Run the following command as the **root** user to check whether the source is valid:

```
apt-get update  //if Ubuntu OS
yum check-update  //if Cent OS
```

If an error is reported during command execution or dependency installation, check whether the network connection is normal, or replace the source in the **/etc/apt/sources.list** file with an available source or use an image source. For details about how to configure a network proxy, see **Configuring a System Network Proxy**.

1.2.5 Installing Dependencies

The following software or dependencies need to be installed for the development kit, Python, auxiliary development tools, and application cross compilation

Parameter	Version
Python	3.7.5
cmake	3.5.1+
protobuf	3.11.3+
gcc	7.3.0 or later

Make/zlib1g/zlib1g-dev/ libsqlite3-dev/openssl/libssl-dev/libffi-dev/unzip/pciutils/net-tools/attrs/psutil/decorator/numpy/scipy/ sympy/cffi/ grpcio/grpcio-tools/requests	There is no version requirement. The version to be installed is subject to the source provided by the OS.
g++-aarch64-linux-gnu	7.3.0 or later

NOTE

- If you install Python and its dependencies as the **root** user, perform following steps.Note that the **sudo** keywords in the commands need to be deleted if any.
- If you install Python and its dependencies as a non-root user, run the **su -username** command to switch to the non-root user first.

Step 1 Install dependencies and GCC software.

Run the following command to install the software.

```
sudo apt-get install -y gcc g++-aarch64-linux-gnu make cmake unzip zlib1g zlib1g-dev
libsqlite3-dev openssl libssl-dev libffi-dev pciutils net-tools  //if Ubuntu OS
sudo yum install -y gcc g++-aarch64-linux-gnu make cmake unzip zlib1g zlib1g-dev
libsqlite3-dev openssl libssl-dev libffi-dev pciutils net-tools  //if Cent OS
```

Run the following commands to check whether the dependencies such as GCC, Make, and Python are installed:

```
gcc --version
make --version cmake --version
aarch64-linux-gnu-g++ --version dpkg -l unzip| grep unzip| grep ii dpkg -l zlib1g|
grep zlib1g| grep ii
dpkg -l zlib1g-dev| grep zlib1g-dev| grep ii
dpkg -l libsqlite3-dev| grep libsqlite3-dev| grep ii dpkg -l openssl| grep openssl|
grep ii
dpkg -l libssl-dev| grep libssl-dev| grep ii dpkg -l libffi-dev|  grep libffi-dev|
grep ii dpkg -l pciutils| grep pciutils| grep ii dpkg -l net-tools| grep net-
tools| grep ii
```

NOTE

If the GCC version is not 7.3.0, see **Update GCC to 7.3.0**

Step 2 Install Python development environment.

Use the following procedure to install Python 3.7.5

1. Download Python and install

```
#using wget to download the Python-3.7.5 zip file, the file will be download to
currently directory
wget https://www.python.org/ftp/python/3.7.5/Python-3.7.5.tgz
# decompress the zip file
tar -zxvf Python-3.7.5.tgz
# enter to the decompress directory
sudo cd Python-3.7.5/
#run the configure command & compile & install
sudo ./configure --prefix=/usr/local/python3.7.5 --enable-shared
```



```
make
sudo make install
```

2. Copy libpython3.7m.so.1.0 dynamic library

```
#copy libpython3.7m.so.1.0 lib to the lib64 library
sudo cp /usr/local/python3.7.5/lib/libpython3.7m.so.1.0 /usr/lib64
#If the /usr/lib64 directory does not exist in the environment, copy the file to the
/usr/lib directory
sudo cp /usr/local/python3.7.5/lib/libpython3.7m.so.1.0 /usr/lib
#When the following information is displayed, enter y to overwrite the
libpython3.7m.so.1.0 file provided by the system.
cp: overwrite 'libpython3.7m.so.1.0'? y
```

3. set the soft links

```
sudo ln -s /usr/local/python3.7.5/bin/python3 /usr/bin/python3.7
sudo ln -s /usr/local/python3.7.5/bin/pip3 /usr/bin/pip3.7
sudo ln -s /usr/local/python3.7.5/bin/python3 /usr/bin/python3.7.5
sudo ln -s /usr/local/python3.7.5/bin/pip3 /usr/bin/pip3.7.5
```

If a message indicating that the link already exists is displayed during the command execution, run the following command to delete the existing link and run the command again:

```
sudo rm -rf  /usr/bin/python3.7.5
sudo rm -rf  /usr/bin/pip3.7.5
sudo rm -rf  /usr/bin/python3.7
sudo rm -rf  /usr/bin/pip3.7
```

4. Check whether install successfully

```
python3.7.5 --version
pip3.7.5 --version
```

If show the correct version number, which means install successfully

Step 3 Install the Python 3 development environment.

If you install Python and its dependencies as a non-root user, add **--user** at the end of each command in this step to ensure that the installation is successful. Example command: **pip3.7.5 install attrs --user**

```
pip3.7.5 install attrs pip3.7.5 install psutil pip3.7.5 install decorator pip3.7.5 install numpy
pip3.7.5 install protobuf==3.11.3
pip3.7.5 install scipy pip3.7.5 install sympy pip3.7.5 install cffi pip3.7.5 install grpcio
pip3.7.5 install grpcio-tools pip3.7.5 install requests
```

During the command execution, if the network connection fails and the message "Could not find a version that satisfies the requirement xxx" is displayed, rectify the fault by referring to configure the pip source.

1.3 Obtaining Ascend Software Packages

Download software packages from Huawei website.The versions of the software packages must be consistent.

Name	Package	OS Architecture	How to Obtain
Toolkit	Ascend-Toolkit-{version}-x86_64- linux_gcc7.3.0.run	Ubuntu X86 CentOS X86	Link
	Ascend-Toolkit-{version}-arm64- linux_gcc7.3.0.run		
Firmware	A500-3000_A500-3010-{version}-Firmware.zip		
Cross	Euler_compile_env_cross.tar.gz		

- {version} indicates the software version, such as 20.0.0.RC1.

2 Install Development Environment

2.1 Install Ascend Toolkit

Step 1 Log in to the installation environment as the installation user of the software package. If you are non-root user, ensure that the current user is the same as the installation dependency user in **Installing Dependencies**.

Step 2 Upload the software packages in **Obtaining Ascend Software Packages** to any directory (for example, **/usr/myupload/**) to your server.

Step 3 Run the install packages

```
# Go to the directory where the software packages are uploaded (for example, /usr/myupload/).
sudo cd /usr/myupload
#install
sudo chmod +x./Ascend-Toolkit-{version}-x86_64-linux_gcc7.3.0.run
./Ascend-Toolkit-{version}-x86_64-linux_gcc7.3.0.run --install
```

- Replace the **{version}** with the actually software version, for example 20.0.0.RC1

The default install path:

/usr/local/Ascend/ascend-toolkit/{version}/x86_64-linux_gcc7.3.0/

Step 4 Set the environment variables

1. Add the environment variables into **~/.bashrc** file and it should be increased based on the actual path of the software you installed.

```
#open the file
vi ~/.bashrc
```

2. Add the following content at the end of the file,and use :wq! command to save it.

```
export PATH=/usr/local/python3.7.5/bin:/usr/local/Ascend/ascend-toolkit/latest/x86_64-
```



```
linux_gcc7.3.0/atc/ccec_compiler/bin:/usr/local/Ascend/ascend-
toolkit/latest/x86_64-linux_gcc7.3.0/atc/bin:$PATH
export PYTHONPATH=/usr/local/Ascend/ascend-toolkit/latest/x86_64-
linux_gcc7.3.0/atc/python/site-packages/te:/usr/local/Ascend/ascend-
toolkit/latest/x86_64-linux_gcc7.3.0/atc/python/site-packages/topi:$PYTHONPATH
export LD_LIBRARY_PATH=/usr/local/Ascend/ascend-toolkit/latest/x86_64-
linux_gcc7.3.0/atc/lib64:$/usr/local/Ascend/add-ons:$LD_LIBRARY_PATH
export ASCEND_OPP_PATH=/usr/local/Ascend/ascend-toolkit/latest/x86_64-
linux_gcc7.3.0/opp
```

3. Apply the change

```
source ~/.bashrc
```

4. Check if the software install success：If show the picture below, which means all the software version and environment variables are correctly

```
atc --help
```

```
root@huawei-2288H-V5:~# atc --help
ATC start working now, please wait for a moment.
atc: usage: ./atc <args>
generate offline model example:
./atc --model=./alexnet.prototxt --weight=./alexnet.caffemodel
--framework=0 --output=./domi
generate offline model for single op example:
./atc --singleop=./op_list.json --output=./op_model
arguments explain:
  --model             Model file
  --singleop          Single op definition file. atc will generate offline model(s) for single op if --singleop is set.
                      Note: Only output, soc_verion, core_type, aicore_num, auto_tune_mode, precision_mode, op_select_implmode, enable_sma
ll_channel, enable_compress_weight, compress_weight_conf enable_single_stream and log are valid in this mode
  --weight            Weight file. Required when framework is Caffe
  --framework         Framework type(0:Caffe; 1:MindSpore; 3:Tensorflow)
  --output            Output file path&name(needn't suffix, will add .om automatically).
                      If --singleop is set, this arg specifies the directory to which the single op offline model will be generated
  --input_shape       Shape of input data. Separate multiple nodes with semicolons (;).Use double quotation marks (") to enclose each argu
ment.E.g.: "input_name1:n1,c1,h1,w1;input_name2:n2,c2,h2,w2"
  --h/help            Show this help message
  --log               Generate atc log. Support debug, info, warning, error, null
  --insert_op_conf    Config file to insert new op
  --op_name_map       Custom op name mapping file
                      Note: A semicolon(;) cannot be included in each path, otherwise the resolved path will not match the expected one.
  --precision_mode    precision mode, support force_fp16, allow_mix_precision, allow_fp32_to_fp16, must_keep_origin_dtype.
  --om                The model file to be converted to json
  --json              The output json file path&name which is converted from a model
```

2.2 Install Cross Compile Tools

There are two ways to install the tools. For Ubuntu OS, recommend to refer to the way 2.2.2 to install.

2.2.1 Use Huawei package to install

Step 1 Install the Ascend toolkit for ARM

- Replace the {version} with the actually software version, for example 20.0.0.RC1

```
# Go to the directory where the software packages are uploaded (for example,
/usr/myupload/).
sudo cd /usr/myupload
#install
sudo chmod +x Ascend-Toolkit-{version}-arm64-linux_gcc7.3.0.run
./Ascend-Toolkit-{version}-arm64-linux_gcc7.3.0.run -install
```

The default install path:

/usr/local/Ascend/ascend-toolkit/{version}/arm64-linux_gcc7.3.0/

Step 2 Install Cross Compile Tools

```
#decompress it to your install path
tar -xzvf Euler_compile_env_cross.tar.gz
```



```
#open bashrc file to set the environment variable
vi ~/.bashrc
#Add the following content at the end of the file,and use :wq! command to save it.
arm_install_path=/usr/myupload/
export
PATH={arm_install_path}/Euler_compile_env_cross/arm/cross_compile/install/bin:{arm_
install_path}/Euler_compile_env_cross/arm/cross_compile/install/aarch64-linux-
gnu/bin:$PATH
#apply the change
source ~/.bashrc
#check version to make sure install success
aarch64-linux-gnu-g++ -v
```

2.2.2 Use apt-get to install (Ubuntu)

Step 1 Install the Ascend toolkit for ARM

- Replace the {version} with the actually software version, for example 20.0.0.RC1

```
# Go to the directory where the software packages are uploaded (for example,
/usr/myupload/).
sudo cd /usr/myupload
#install
sudo chmod +x Ascend-Toolkit-{version}-arm64-linux_gcc7.3.0.run
./Ascend-Toolkit-{version}-arm64-linux_gcc7.3.0.run -install
```

The default install path:

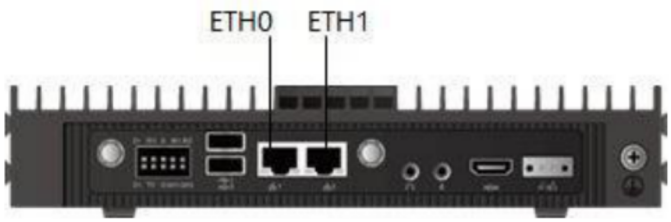
/usr/local/Ascend/ascend-toolkit/{version}/arm64-linux_gcc7.3.0/

Step 2 Install Cross Compile Tools

```
#use apt-get to install from internet
apt-get install aarch64-linux-gnu-g++
#check version to make sure install success
aarch64-linux-gnu-g++ -v
```

2.3 Configure Atlas 500

Atlas Network Ports



Default Information

Category	Name	Default Value

CLI login data of the Atlas 500 IES	Initial user name and password	• Default user name: admin • Default password: Huawei12#$
	Default password in develop mode	• Default password: Huawei@SYS3
WebUI login data of the Atlas 500 IES	Initial user name and password	• Default user name: admin • Default password: Huawei12#$
Network port data of the Atlas 500 IES	Initial IP address of the management network port	• Default IP address of ETH0: 192.168.2.111, corresponding to network port 1 of the device hardware • Default IP address of ETH1: 192.168.3.111, corresponding to network port 2 of the device hardware

Step 1 Upgrade Atlas 500

Log in Atlas 500 WebUI: https://XXX.XXX.XXX.XXX.

NOTE

- *XXX.XXX.XXX.XXX* is the default IP address of Atlas 500 ETH0 or ETH1 which you used.
- If a message indicating a security certificate problem of the website is displayed, click Continue to this website (not recommended).
- (option)You can change the initial user name and password, change the initial IP address, set the system time, check health status, etc.

Refer to the upgrade guide to upgrade

Step 2 Open the sftp and enter the develop mode

Use terminal (for example MobaXterm) connect to Atlas 500. The default user name\password\ip address refer to Default Information. Use following command to open the SFTP and enter the develop mode.

```
#open SFTP
sftp enable
#enter develop mode
Develop
#enter the develop mode password
Huawei@SYS3
```

Step 3 Check the FTP work.

After you enter the develop mode, you can get the following result: from IES to "Euler"


```
IES:/->sftp enable
set enable success.
IES:/->develop
Tips:The root password is an initial password. Change it for security purposes.
input root passwd
Password:
Last login: Fri Sep 18 14:45:26 UTC 2020 on pts/0
Euler:~ #
```

Use the terminal tool try to upload a file to Atlas 500.

You can upload the file to the /tmp directory only.

```
#enter the /tmp/ directory
cd /tmp
#create your dir, the {yourdir} is the dir name you want to create.
mkdir {yourdir}
#get the right for upload
chmod 777 {yourdir}
```

NOTE

Some terminal tools need to reconnect after you open the sftp.

The directory "/tmp" only work on the memory, if you reboot Atlas 500, all the files will be deleted. You can copy all the files to "/var/lib/docker/{your dir}" to make sure the files work on disk.

```
cp -r /tmp/{your dir} /var/lib/docker/{your dir}
```


3 Compile and Run a Sample

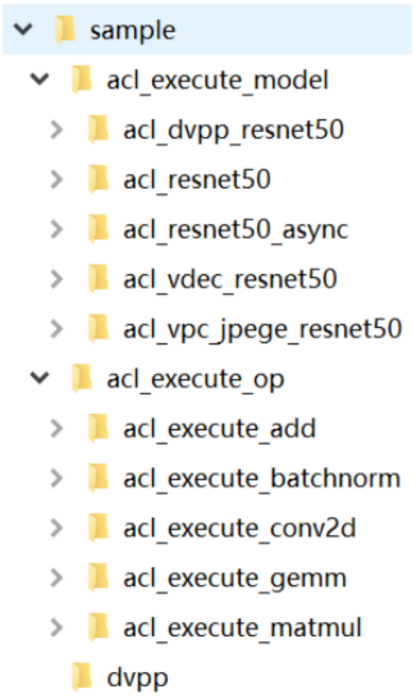
3.1 Compile a Sample

After install the ascend toolkit, there are some sample projects which developed by C++ in Following path on the server:

/usr/local/Ascend/ascend-toolkit/{version}/arm64-linux_gcc7.3.0/acllib/sample/

The details of the sample projects, you can refer to the "ACL Sample Description" section of 《Application Software Development Guide》

Following is the directory structure:



This section will use the project: "acl_vpc_jpege_resnet50" to finish the compile and run it on Atlas 500.

Step 1 Obtain the model

Obtain the weight file (*.caffemodel) of the ResNet-50 network from the https://gitee.com/HuaweiAscend/models/tree/master/computer_vision/classification/resnet50 and upload the file to the **acl_vpc_jpege_resnet50/caffe_model** directory.

Step 2 Convert the model

Convert the ResNet-50 network to an offline model adapted to the Ascend AI Processor (an *.om file). In this project, you need to set CSC parameters to convert YUV420SP images to RGB images during model conversion. For more information of the ATC tool, the "Auxiliary Development Tools> ATC Tool Instructions" section of 《Application Software Development Guide》

```
#Go to the acl_vpc_jpege_resnet50 directory
cd /usr/local/Ascend/ascend-toolkit/{version}/arm64-
linux_gcc7.3.0/acllib/sample/acl_execute_model/acl_vpc_jpege_resnet50/
#run ATC tools to convert the model
atc --model=caffe_model/resnet50.prototxt --weight=caffe_model/resnet50.caffemodel
--framework=0 --output=model/resnet50_aipp --soc_version=Ascend310 --
insert_op_conf=caffe_model/aipp.cfg
```

After the ACT tools run success, you can find the file "**resnet50_aipp.om**" in the "**model**" directory. And you can see infotmation: "ATC run success, welcome to the next use."

```
[root@localhost acl_vpc_jpege_resnet50]# atc --model=caffe_model/resnet50.prototxt --weight=caffe_mod
el/resnet50.caffemodel --framework=0 --output=model/resnet50_aipp --soc_version=Ascend310 --insert_op
_conf=caffe_model/aipp.cfg
ATC start working now, please wait for a moment.
ATC run success, welcome to the next use.
```

Step 3 Set environment variables

In this project, the build script **src/CMakeLists.txt** builds code based on the header file path and library file path set by environment variables.

1. open **~/.bashrc** file.

```
#open the file
vi ~/.bashrc
```

2. Add the following content at the end of the file, and use :wq! command to save it.

```
export DDK_PATH= /usr/local/Ascend/ascend-toolkit/latest/arm64-linux_gcc7.3.0
export NPU_HOST_LIB= /usr/local/Ascend/ascend-toolkit/latest/arm64-
linux_gcc7.3.0/acllib/lib64/stub
```

3. Apply the change

```
source ~/.bashrc
```

Step 4 Compile the source code

Run the following command:

```
#Go to the acl_vpc_jpege_resnet50 directory
cd /usr/local/Ascend/ascend-toolkit/{version}/arm64-
linux_gcc7.3.0/acllib/sample/acl_execute_model/acl_vpc_jpege_resnet50/
# create a directory for storing build outputs. For example, the directory created
in this sample is build/intermediates/host
mkdir -p build/intermediates/host
#Go to the build/intermediates/host directory and run the cmake command to cross
compile the source code.
```



```
#../../../src indicates the directory where the CMakeLists.txt file is located.
Replace it with the actual path
cd build/intermediates/host
cmake ../../../src -DCMAKE_CXX_COMPILER=aarch64-linux-gnu-g++ -
DCMAKE_SKIP_RPATH=TRUE
make
```

After the compile success, you can find the file "**main**" in the "**out**" directory. And you can see infotmation: "[100%] Built target main"

```
[root@localhost host]# make
Scanning dependencies of target main
[ 16%] Building CXX object CMakeFiles/main.dir/utils.cpp.o
[ 33%] Building CXX object CMakeFiles/main.dir/dvpp_process.cpp.o
[ 50%] Building CXX object CMakeFiles/main.dir/model_process.cpp.o
[ 66%] Building CXX object CMakeFiles/main.dir/sample_process.cpp.o
[ 83%] Building CXX object CMakeFiles/main.dir/main.cpp.o
[100%] Linking CXX executable /usr/local/Ascend/ascend-toolkit/20.0.0.RC1/arm64-linux_gcc7.3.0/acllib
/sample/acl_execute_model/acl_vpc_jpege_resnet50/out/main
[100%] Built target main
```

3.2 Run the Sample on Atlas 500

Login to Atlas 500 and enter "develop mode".

Package all the sample executable files and its dependcies. In this sample project must package the "out"、 "data"、"model" directory and the "acl.json" file.

```
#copy all the files to a temporary directory, for example "outfiles".
cd /usr/local/Ascend/ascend-toolkit/{version}/arm64-
linux_gcc7.3.0/acllib/sample/acl_execute_model/acl_vpc_jpege_resnet50/
mkdir outfiles
cp -r ./data ./outfiles
cp -r ./model ./outfiles
cp -r ./out ./outfiles
mkdir ./outfiles/src
cp ./src/alc.json ./outfiles/src
#compress all the files
tar -zcvf outfiles.tar ./outfiles
```

Upload the package file "outfiles.tar" to any directory and decompress all the files. For how to login and upload, you can refer to: 2.3 Configure Atlas 500

The following command to run the sample file.

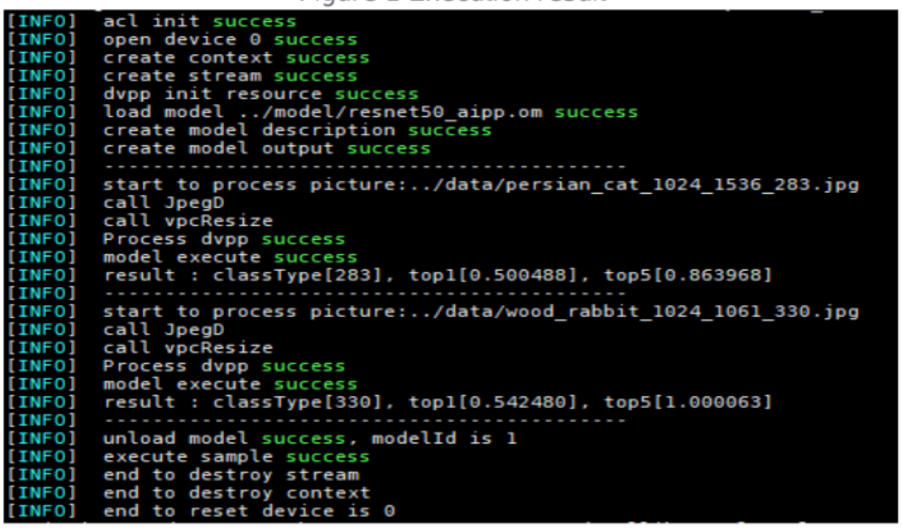
```
#go to the directory where the "outfiles.tar" file located
cd {your dir}
#decompress the file
tar -xzf outfiles.tar
#go to the directory where the "main" file located
cd outfiles/out
#add the executable to "main"
chmod +x main
#run
./main 0
./main 1
./main 2
./main 3
```


1. Parameter 0 execution result

Decode two *.jpg images into two YUV420SP NV12 images, resize them, and perform model inference to obtain the inference results of the two images.

```
./main 0
```

Figure 1 Execution result

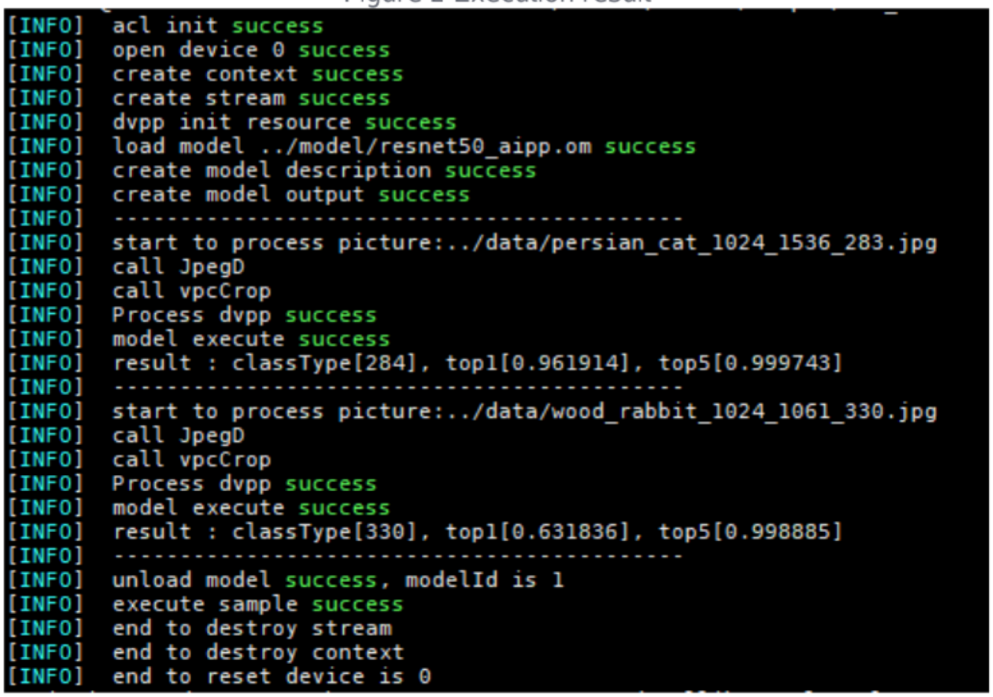


2. Parameter 1 execution result

Decode two *.jpg images into two YUV420SP NV12 images, crop them, and perform model inference to obtain the inference results of the two images.

```
./main 1
```

Figure 2 Execution result

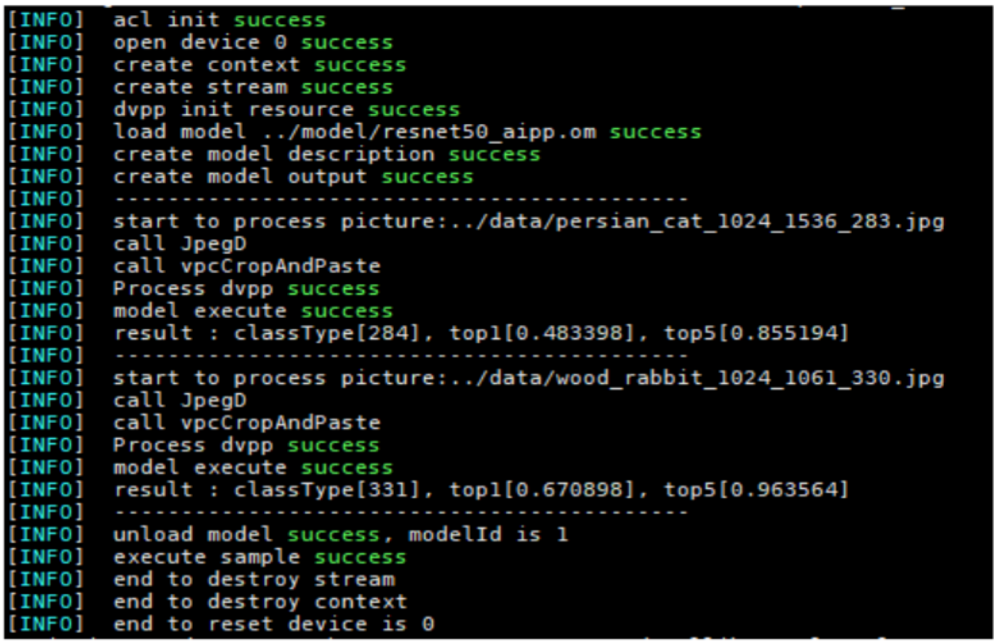


3. Parameter 2 execution result

Decode two *.jpg images into two YUV420SP NV12 images, crop and paste them, and perform model inference to obtain the inference results of the two images.

```
./main 2
```

Figure 3 Execution result

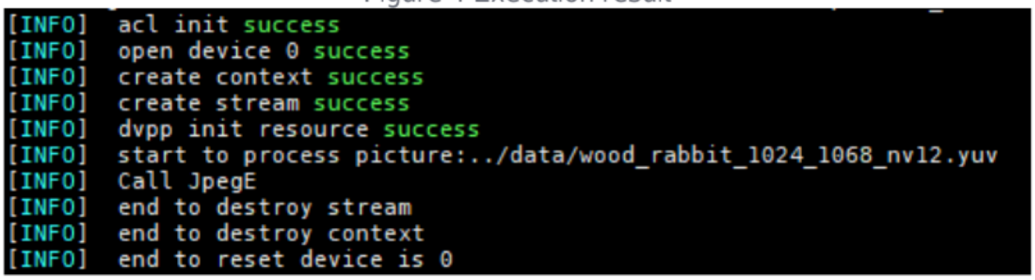


4. Parameter 3 execution result

Encode a YUV420SP image into a *.jpg image.

```
./main 3
```

Figure 4 Execution result



3.3 Use Third lib Compile

If you used a third library in your application, for example FFmpeg etc. you need to get the ARM version lib of the software (from web or get the source code to compile).Here is a sample for how to compile the ARM version libs of the FFmpeg (refer to Compile FFmpeg).

Step 1 Check cross compile tools

Make sure the cross compile tools have been installed. Refer to 2.2 Install Cross Compile Tools

```
#check version to make sure install success
aarch64-linux-gnu-g++ -v
```

Step 2 Get the source code

Download the source code from internet: http://ffmpeg.org/download.html#releases, for example ffmpeg-4.2.4.tar.bz2, then upload the file to any directory in the server.

Step 3 Compile and install

```
# go to the directory where the FFmpeg source code file located
cd {your dir}
#decompress (my version is 4.2.4)
tar -xjvf ffmpeg-4.2.4.tar.bz2
cd ffmpeg-4.2.4
#use aarch64-linux-gnu-g++ to compile
./configure --prefix=<where your ffmpeg arm install place> --target-os=linux --
arch=aarch64 --enable-cross-compile --cross-prefix=<your aarch64-linux-gnu
path>/aarch64-linux-gnu- --enable-shared --disable-doc
make && make install
```

NOTE

<your aarch64-linux-gnu path> is the path where the **aarch64-linux-gnu-g++** in your cross compile tools directory. If you use Huawei package to install your corss compile tools, the path as below:

/usr/{your dir}/Euler_compile_env_cross/arm/cross_compile/install/bin/

you can run following command to find the actual path

```
find / -name aarch64-linux-gnu-g++
```

Step 4 Package the libs

When the build is complete, you can find three directories: ” bin”、” lib”、” include” in your ffmpeg install place.

```
#go to your ffmpag install directory
cd {your ffmpeg dir}
tar -zcvf ffmpeg.tar ./lib
```

Step 5 Use the libs

1. Log in Atlas 500(develop mode) and Upload the ffmpeg.tar.
2. Decompress the file.

```
#go to directory where the ffmpeg.tar uploaded
cd {your dir}
```



```
#decompress the file
tar -xzf ffmpeg.tar
```

3. Set environment variables

open **~/.bashrc** file => Add the content at the end of the file, and use :wq! command to save it.=>apply the change

```
#open the file
vi ~/.bashrc
#add the content at the end of the file,and use :wq to save, replace the {your dir}
with the actual path
export LD_LIBRARY_PATH={your dir}/lib/:$LD_LIBRARY_PATH
#Apply the change
source ~/.bashrc
```


4 Other Reference

4.1 Configuring a System Network Proxy

Step 1 Log in to the user environment as the **root** user.

Open the **/etc/profile** file:

```
vi /etc/profile
```

Step 2 Add the following content to the file, save the file, and exit:

```
export http_proxy="http://user:password@proxyserverip:port"
export https_proxy="http://user:password@proxyserverip:port"
```

In the preceding commands, **user** indicates the username on the intranet, **password** indicates the user password, **proxyserverip** indicates the IP address of the proxy server, and **port** indicates the port number.

Step 3 make the configuration take effect.

```
source /etc/profile
```

4.2 Cent OS Update GCC to 7.3.0

Log in to the sever then Run following command

```
#If Cent OS
yum -y install centos-release-scl
yum -y install devtoolset-7-gcc devtoolset-7-gcc-c++ devtoolset-7-binutils
scl enable devtoolset-7 bash
echo "source /opt/rh/devtoolset-7/enable" >>/etc/profile
```

4.3 Configure the pip source

Step 1 Go to the .pip directory

```
cd ~/.pip
```

If a message indicating that the directory does not exist is displayed, run the following command to create the directory:

```
mkdir ~/.pip
cd ~/.pip
```

Step 2 Create a **pip.conf** file in the **.pip** directory:


```
touch pip.conf
```

Step 3 Edit the **pip.conf** file.

Run the **vi pip.conf** command to open the **pip.conf** file and edit the file as follows. You can replace the address to any other trust address .

```
[install]
#Configure the trusted host as required.
trusted-host=mirrors.tools.huawei.com
[global]
#Configure the sources as required.
index-url=http://mirrors.tools.huawei.com/pypi/simple/
```

Run the **:wq!** command to save the file and exit.

