# sophon-stream environment installation guide
## Table of contents

The environments Sophon Stream relies on mainly include the TPU-NNTC and TPU-MLIR environments for compiling and quantizing models, the development environment for compiling C++ programs, and the running environment for deploying programs.

## 1 TPU-MLIR environment construction
If you use the BM1684X processor, it is recommended to compile BModel with TPU-MLIR. It is usually necessary to install the TPU-MLIR environment on the x86 host. The x86 host has Ubuntu16.04/18.04/20.04 system installed and the running memory is more than 12GB. The TPU-MLIR environment installation steps mainly include:

1. Install Docker

   If docker is already installed, please skip this section.
    ```bash
    #Install docker
    sudo apt-get install docker.io
    # Docker command can be executed without root privileges
    # Create a docker user group. If there is already a docker group, an error will be reported. It doesn't matter and can be ignored.
    sudo groupadd docker
    # Add the current user to the docker group
    sudo usermod -aG docker $USER
    # Switch the current session to a new group or log in again to restart the X session
    newgrp docker
    ```
    > **Tips**: You need to logout the system and then log in again. You do not need sudo when using docker.

2. Download and unzip TPU-MLIR

    Download the TPU-MLIR compressed package from [Computing Energy Official Website](https://developer.sophgo.com/site/index/material/31/all.html) and name it such as tpu-mlir_vx.yz-hash-date. tar.gz, xyz represents the version number, and decompress it.
    ```bash
    ```

```
    tar zxvf tpu-mlir_vx.yz-<hash>-<date>.tar.gz
```

3. Create and enter docker

    The docker used by TPU-MLIR is sophgo/tpuc_dev:2.2. The docker image is bound to tpu-mlir. In rare cases, tpu-mlir may be updated and a new image is required.

```bash
    # If the current system does not have a corresponding image, it will be automatically downloaded from docker hub.
    # Here, this level directory is mapped to the /workspace directory in docker. Users need to map the demo directory to docker according to the actual situation.
    # myname is just an example of a name, please specify it as the name of the container you want.
    docker run --name myname -v $PWD:/workspace -it sophgo/tpuc_dev:v2.2
    # At this time, you have entered docker and are in the /workspace directory.
    # Initialize software environment
    cd /workspace/tpu-mlir_vx.yz-<hash>-<date>
    source ./envsetup.sh
```

This image is only used to compile and quantify the model. Please compile and run the program in the development and running environments. For more tutorials on TPU-MLIR, please refer to the "TPU-MLIR Quick Start Manual" and "TPU- MLIR Development Reference Manual.

## 2 TPU-NNTC environment construction
If you use BM1684 processor, it is recommended to use TPU-NNTC to compile BModel. It is usually necessary to install the TPU-NNTC environment on the x86 host. The x86 host has Ubuntu16.04/18.04/20.04 system installed and the running memory is more than 12GB. The TPU-NNTC environment installation steps mainly include:

1. Install Docker

   If docker is already installed, please skip this section.

```bash
    #Install docker
    sudo apt-get install docker.io
    # Docker command can be executed without root privileges
    # Create a docker user group. If there is already a docker group, an error will be reported. It doesn't matter and can be ignored.
    sudo groupadd docker
    # Add the current user to the docker group
    sudo usermod -aG docker $USER
    # Switch the current session to a new group or log in again to restart the X session
    newgrp docker
```

> **Tips**: You need to logout the system and then log in again. You do not need sudo when using docker.

2. Download and unzip TPU-NNTC

    Download the TPU-NNTC compressed package from [Computing Energy Official Website](https://developer.sophgo.com/site/index/material/28/all.html) and name it such as tpu-nntc_vx.yz-hash-date. tar.gz, xyz represents the version number, and decompress it.

```bash
    mkdir tpu-nntc
    # Extract the compressed package to tpu-nntc
    tar zxvf tpu-nntc_vx.yz-<hash>-<date>.tar.gz --strip-components=1 -C tpu-nntc
```

3. Create and enter docker

    The docker used by TPU-NNTC is sophgo/tpuc_dev:2.1. The docker image is bound to tpu-nntc. In a few cases, tpu-nntc may be updated and a new image is required.
    ```bash
    cd tpu-nntc
    # Enter docker. If there is no corresponding image in the current system, it will be automatically downloaded from docker hub.
    # Here, the upper-level directory of tpu-nntc is mapped to the /workspace directory in docker. The user needs to map the demo directory to docker according to the actual situation.
    # The 8001 to 8001 port mapping is used here, which will be used later when using the ufw visualization tool.
    # If the port is already occupied, please replace it with another unoccupied port and make adjustments later as needed.
    docker run --name myname -v $PWD/..:/workspace -p 8001:8001 -it sophgo/tpuc_dev:v2.1
    # At this time, you have entered docker and are in the /workspace directory.
    # Initialize the software environment below
    cd /workspace/tpu-nntc
    source scripts/envsetup.sh
    ```
This image is only used to compile and quantify the model. Please compile and run the program in the development and running environments. For more tutorials on TPU-NNTC, please refer to the "TPU-NNTC Quick Start Guide" and "TPU- NNTC Development Reference Manual".

## 3 Development and operation environment construction of x86 PCIe platform
If you install a PCIe accelerator card on the x86 platform, the development environment and operating environment can be unified, and you can build the development and operating environments directly on the host.

### 3.1 Install libsophon
Download the libsophon installation package from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html), including:
* sophon-driver_x.y.z_amd64.deb
* sophon-libsophon_x.y.z_amd64.deb
* sophon-libsophon-dev_x.y.z_amd64.deb

Among them: xyz represents the version number; sophon-driver contains the PCIe accelerator card driver; sophon-libsophon contains the runtime environment (library files, tools, etc.); sophon-libsophon-dev contains the development environment (header files, etc.). If you are only installing on a deployment environment, you do not need to install sophon-libsophon-dev.
```bash
#Install dependent libraries, only need to be executed once
sudo apt install dkms libncurses5
# Install libsophon
sudo dpkg -i sophon-*amd64.deb
# Execute the following command in the terminal, or log out and log in as the current user to use bm-smi and other commands:
source /etc/profile
```

For more information about libsophon, please refer to "LIBSOPHON User Manual.pdf".

### 3.2 Install sophon-ffmpeg and sophon-opencv
Download the sophon-mw installation package from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html), including:
* sophon-mw-sophon-ffmpeg_x.y.z_amd64.deb

* sophon-mw-sophon-ffmpeg-dev_x.y.z_amd64.deb
* sophon-mw-sophon-opencv_x.y.z_amd64.deb
* sophon-mw-sophon-opencv-dev_x.y.z_amd64.deb

Among them: xyz represents the version number; sophon-ffmpeg/sophon-opencv contains the ffmpeg/opencv runtime environment (library files, tools, etc.); sophon-ffmpeg-dev/sophon-opencv-dev contains the development environment (header files, pkgconfig , cmake, etc.). If you are only installing on the deployment environment, you do not need to install sophon-ffmpeg-dev/sophon-opencv-dev.

sophon-mw-sophon-ffmpeg depends on the sophon-libsophon package, and sophon-mw-sophon-opencv depends on sophon-mw-sophon-ffmpeg, so the installation sequence must
Install libsophon first, then sophon-mw-sophon-ffmpeg, and finally sophon-mw-sophon-opencv.

If the libstdc++ library used in the running environment uses an old version of the ABI interface before GCC5.1 (typically a CENTOS system), please use the sophon-mw-sophon-opencv-abi0 related installation package.

```bash
# Install sophon-ffmpeg
sudo dpkg -i sophon-mw-sophon-ffmpeg_*amd64.deb sophon-mw-sophon-ffmpeg-dev_*amd64.deb
# Install sophon-opencv
sudo dpkg -i sophon-mw-sophon-opencv_*amd64.deb sophon-mw-sophon-opencv-dev_*amd64.deb
# Execute the following command in the terminal, or logout and then log in as the current user to use the installed tools
source /etc/profile
```

For more information about sophon-mw, please refer to "MULTIMEDIA User Manual.pdf" and "MULTIMEDIA Development Reference Manual.pdf".

### 3.3 Compile and install sophon-sail
If the routine depends on sophon-sail, you need to compile and install sophon-sail, otherwise you can skip this chapter. You need to download the compressed package of sophon-sail from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html), named like sophon-sail_x.yztar.gz, xyz Represents the version number.
1. Unzip and enter the directory

    ```bash
    tar zxvf sophon-sail_x.yztar.gz
    cd sophon-sail
    ```

2. Create the compilation folder build and enter the build folder

    ```bash
    mkdir build && cd build
    ```

3. Execute the compilation command

    ```bash
    cmake..
    make
    ```

4. Install the SAIL dynamic library and header files. The compilation results will be installed under `/opt/sophon`

```bash
    sudo make install
    ```

5. Package and generate python wheel. The path of the generated wheel package is
`python/pcie/dist`

    ```bash
    cd ../python/pcie
    chmod +x sophon_pcie_whl.sh
    ./sophon_pcie_whl.sh
    ```

6. Install python wheel

    ```bash
    # The file name needs to be modified according to the actual generated wheel
package.
    pip3 install ./dist/sophon-*-py3-none-any.whl --force-reinstall
    ```

## 4 Development and operation environment construction of SoC platform
For the SoC platform, after installing SophonSDK (>=v22.09.02), the
corresponding libsophon, sophon-opencv and sophon-ffmpeg runtime library
packages have been integrated internally. They are located under `/opt/sophon/`
and can be directly used in the running environment. . The program is usually
cross-compiled on the x86 host to enable it to run on the SoC platform. For the
SophonSDK firmware refresh method, please refer to [FAQ document] (./FAQ.md#12-
How to use sd card to refresh firmware in soc mode).

### 4.1 Cross-compilation environment construction
You need to use SOPHON SDK to build a cross-compilation environment on the x86
host, and package the header files and library files that the program depends on
into the soc-sdk directory.
1. Install the cross-compilation tool chain
    ```bash
    sudo apt-get install gcc-aarch64-linux-gnu g++-aarch64-linux-gnu
    ```
    If an error is reported: `/lib/aarch64-linux-gnu/libc.so.6: version
'GLIBC_2.33' not found`.
    This is because the cross-compilation toolchain version on your host is too
high. You can reinstall it with the following command:
    ```bash
    sudo apt remove cpp-*-aarch64-linux-gnu
    sudo apt-get install gcc-7-aarch64-linux-gnu g++-7-aarch64-linux-gnu
    sudo ln -s /usr/bin/aarch64-linux-gnu-gcc-7 /usr/bin/aarch64-linux-gnu-gcc
    sudo ln -s /usr/bin/aarch64-linux-gnu-g++-7 /usr/bin/aarch64-linux-gnu-g++
    ```

2. Package libsophon

    Download the sophon-img installation package from [Sophgo official website]
(https://developer.sophgo.com/site/index/material/28/all.html), which includes
libsophon_soc_x.y.z_aarch64.tar.gz, xyz Indicates the version number and
decompresses it.

    ```bash
    #Create the root directory of dependent files
    mkdir -p soc-sdk
    # Unzip libsophon_soc_x.y.z_aarch64.tar.gz
    tar -zxf libsophon_soc_${xyz}_aarch64.tar.gz
    ```

```bash
    # Copy the relevant library directory and header file directory to the root directory of the dependent file
    cp -rf libsophon_soc_${xyz}_aarch64/opt/sophon/libsophon-${xyz}/lib ${soc-sdk}
    cp -rf libsophon_soc_${xyz}_aarch64/opt/sophon/libsophon-${xyz}/include ${soc-sdk}
```

3. Package sophon-ffmpeg and sophon-opencv

    Download the sophon-mw installation package from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html), which includes sophon-mw-soc_x.y.z_aarch64.tar .gz, xyz represents the version number, and decompress it.

```bash
    # Unzip sophon-mw-soc_x.y.z_aarch64.tar.gz
    tar -zxf sophon-mw-soc_${xyz}_aarch64.tar.gz
    # Copy the library directory and header file directory of ffmpeg and opencv to the soc-sdk directory
    cp -rf sophon-mw-soc_${xyz}_aarch64/opt/sophon/sophon-ffmpeg_${xyz}/lib ${soc-sdk}
    cp -rf sophon-mw-soc_${xyz}_aarch64/opt/sophon/sophon-ffmpeg_${xyz}/include ${soc-sdk}
    cp -rf sophon-mw-soc_${xyz}_aarch64/opt/sophon/sophon-opencv_${xyz}/lib ${soc-sdk}
    cp -rf sophon-mw-soc_${xyz}_aarch64/opt/sophon/sophon-opencv_${xyz}/include ${soc-sdk}
```

Here, the cross-compilation environment has been set up. Next, you can use the packaged soc-sdk to compile the programs that need to be run on the SoC platform. For more cross-compilation information, please refer to "LIBSOPHON User Manual.pdf".

### 4.2 Cross-compile and install sophon-sail
If the routine depends on sophon-sail, you need to compile and install sophon-sail, otherwise you can skip this chapter. Sophon-sail needs to be cross-compiled on the x86 host and installed on the SoC platform.

You need to download the compressed package of sophon-sail from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html), named like sophon-sail_x.yztar.gz, xyz Represents the version number.
1. Unzip and enter the directory

```bash
    tar zxvf sophon-sail_x.yztar.gz
    cd sophon-sail
```

2. Create the compilation folder build and enter the build folder

```bash
    mkdir build && cd build
```

3. Execute the compilation command

    Use the specified python3 and cross-compile to compile SAIL containing bmcv, sophon-ffmpeg, and sophon-opencv. The installation method of python3 can be obtained from the official python documentation, or from [this link](http://219.142 .246.77:65000/sharing/8MlSKnV8x) to download the compiled python3. The python3 path used in this example is `python_3.8.2/bin/python3`, and the dynamic library directory of python3 is `python_3.8.2/lib`.

Please refer to [4.1 Cross-compilation environment construction] (#41-Cross-compilation environment construction) to obtain the cross-compilation library packages of libsophon, sophon-ffmpeg and sophon-opencv.

```bash
# Please modify the paths of DPYTHON_EXECUTABLE, DCUSTOM_PY_LIBDIR, DLIBSOPHON_BASIC_PATH and DOPENCV_BASIC_PATH according to the actual situation.
 cmake -DBUILD_TYPE=soc \
    -DCMAKE_TOOLCHAIN_FILE=../cmake/BM168x_SOC/ToolChain_aarch64_linux.cmake \
    -DPYTHON_EXECUTABLE=python_3.8.2/bin/python3 \
    -DCUSTOM_PY_LIBDIR=python_3.8.2/lib \
    -DLIBSOPHON_BASIC_PATH=libsophon_soc_${xyz}_aarch64/opt/sophon/libsophon-${xyz} \
    -DFFMPEG_BASIC_PATH=sophon-mw-soc_${xyz}_aarch64/opt/sophon/sophon-ffmpeg_${xyz} \
    -DOPENCV_BASIC_PATH=sophon-mw-soc_${xyz}_aarch64/opt/sophon/sophon-opencv_${xyz} \
    ..
make
```
Compile parameters:

* BUILD_TYPE: The type of compilation. There are currently two modes: pcie and soc. pcie is used to compile SAIL packages available on the x86 host. soc means using cross-compilation to compile SAIL packages available on the soc on the x86 host. Default pcie.

* ONLY_RUNTIME: Whether the compilation result only contains runtime, not bmcv, sophon-ffmpeg, sophon-opencv. If this compilation option is `ON`, the SAIL codec and Bmcv interfaces are not available, and only the inference interface is available. Default `OFF`.

* INSTALL_PREFIX: The installation path when executing make install. The default is `/opt/sophon` in pcie mode, which is consistent with the installation path of libsophon. The default is `build_soc` in cross-compilation mode.

* PYTHON_EXECUTABLE: The path name (path + name) of python3 used for compilation. By default, the default python3 in the current system is used.

* CUSTOM_PY_LIBDIR: The path to the python3 dynamic library used for compilation (only includes the path). By default, the default python3 dynamic library directory in the current system is used.

* LIBSOPHON_BASIC_PATH: In cross-compilation mode, the path of libsophon. If the configuration is incorrect, the compilation will fail. This compilation option does not take effect in pcie mode.

* FFMPEG_BASIC_PATH: In cross-compilation mode, the path of sophon-ffmpeg will fail to compile if the configuration is incorrect and ONLY_RUNTIME is `ON`. This compilation option does not take effect in pcie mode.

* OPENCV_BASIC_PATH: In cross-compilation mode, the path of sophon-opencv will fail to compile if the configuration is incorrect and ONLY_RUNTIME is `ON`. This compilation option does not take effect in pcie mode.

4. Install the SAIL dynamic library and header files. The compilation results will be installed under `../build_soc`

```bash
sudo make install
```

    Copy `sophon-sail` under the `build_soc` folder to the `/opt/sophon` directory of the target SOC, and then return to the compilation machine for subsequent operations.

5. Package and generate python wheel. The path of the generated wheel package is `python/soc/dist`

    ```bash
    cd ../python/soc
    chmod +x sophon_soc_whl.sh
    ./sophon_soc_whl.sh
    ```

5. Install python wheel

    Copy the generated wheel package to the target SOC, and then execute the following installation command
    ```bash
    # The file name needs to be modified according to the actual generated wheel package, xyz represents the version number
    pip3 install sophon_arm-*-py3-none-any.whl --force-reinstall
    ```

## 5 arm PCIe platform development and operating environment construction
If you install a PCIe accelerator card on the arm platform, the development environment and operating environment can be unified, and you can build the development and operating environments directly on the host.
Here are the environment installation methods for Galaxy Kirin v10 machines. For details on other types of machines, please refer to the official website development manual.
### 5.1 Install libsophon
Download the libsophon installation package from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html),
The installation package consists of a file, where "$arch" is the hardware architecture of the current machine. Use the following command to obtain the arch of the current server:
```
uname -m
```
Usually the hardware architecture corresponding to x86_64 machines is x86_64, and the hardware architecture corresponding to arm64 machines is aarch64:
```
libsophon_x.y.z_$arch.tar.gz, xyz represents the version number
```
It can be installed through the following steps:

**Note: If there is an old version, please refer to the uninstallation steps below to uninstall the old version. **
```
tar -xzvf libsophon_${xyz}_aarch64.tar.gz
sudo cp -r libsophon_${xyz}_aarch64/* /
sudo ln -s /opt/sophon/libsophon-${xyz} /opt/sophon/libsophon-current
```
Next, please set up the driver compilation environment according to the requirements of the Linux distribution you are using, and then do the following:
```
sudo ln -s /opt/sophon/driver-${xyz}/$bin /lib/firmware/bm1684x_firmware.bin
sudo ln -s /opt/sophon/driver-${xyz}/$bin /lib/firmware/bm1684_ddr_firmware.bin
sudo ln -s /opt/sophon/driver-${xyz}/$bin /lib/firmware/bm1684_tcm_firmware.bin
cd /opt/sophon/driver-${xyz}
```
Here "$bin" is the full name of the bin file with the version number. For the bm1684x board, it is a53lite_pkg.bin. For the bm1684 board, it is such as

bm1684_ddr.bin_v3.1.1-63a8614d-220906 and bm1684_tcm.bin_v3.1.1-63a8614d - 220906.

After that, you can compile the driver (this does not depend on dkms):
```
sudo make SOC_MODE=0 PLATFORM=asic SYNC_API_INT_MODE=1 \
          TARGET_PROJECT=sg_pcie_device FW_SIMPLE=0 \
          PCIE_MODE_ENABLE_CPU=1
sudo cp ./bmsophon.ko /lib/modules/$(uname -r)/kernel/
sudo depmod
sudo modprobe bmsophon
```
Finally some configuration work:

Add library and executable paths:
```
sudo cp /opt/sophon/libsophon-current/data/libsophon.conf /etc/ld.so.conf.d/
sudo ldconfig
sudo cp /opt/sophon/libsophon-current/data/libsophon-bin-path.sh /etc/profile.d/
```
Execute the following commands in the terminal, or log out and log in as the current user to use bm-smi and other commands:
```
source /etc/profile
```
Add cmake config file:
```
sudo mkdir -p /usr/lib/cmake/libsophon
sudo cp /opt/sophon/libsophon-current/data/libsophon-config.cmake
/usr/lib/cmake/libsophon/
```
Uninstall method:
```
sudo rm -f /etc/ld.so.conf.d/libsophon.conf
sudo ldconfig
sudo rm -f /etc/profile.d/libsophon-bin-path.sh
sudo rm -rf /usr/lib/cmake/libsophon
sudo rmmod bmsophon
sudo rm -f /lib/modules/$(uname -r)/kernel/bmsophon.ko
sudo depmod
sudo rm -f /lib/firmware/bm1684x_firmware.bin
sudo rm -f /lib/firmware/bm1684_ddr_firmware.bin
sudo rm -f /lib/firmware/bm1684_tcm_firmware.bin
sudo rm -f /opt/sophon/libsophon-current
sudo rm -rf /opt/sophon/libsophon-0.4.6
sudo rm -rf /opt/sophon/driver-0.4.6
```
For other platform machines, please refer to [libsophon installation tutorial](https://doc.sophgo.com/sdk-docs/v22.12.01/docs_latest_release/docs/libsophon/guide/html/1_install.html).
For more information about libsophon, please refer to "LIBSOPHON User Manual.pdf"

### 5.2 Install sophon-ffmpeg and sophon-opencv
Download the sophon-mw installation package from [Sophgo official website](https://developer.sophgo.com/site/index/material/28/all.html),
The installation package consists of one file:
```
sophon-mw_x.y.z_aarch64.tar.gz, xyz represents the version number
```
It can be installed through the following steps:

First install the libsophon package according to the "LIBSOPHON User Manual", and then,

```
tar -xzvf sophon-mw_${xyz}_aarch64.tar.gz
sudo cp -r sophon-mw_${xyz}_aarch64/* /
sudo ln -s /opt/sophon/sophon-ffmpeg_${xyz} /opt/sophon/sophon-ffmpeg-latest
sudo ln -s /opt/sophon/sophon-opencv_${xyz} /opt/sophon/sophon-opencv-latest
sudo ln -s /opt/sophon/sophon-sample_${xyz} /opt/sophon/sophon-sample-latest
sudo sed -i "s/usr\/local/opt\/sophon\/sophon-ffmpeg-latest/g" /opt/sophon/sophon-ffmpeg-latest/lib/pkgconfig/*.pc
sudo sed -i "s/^prefix=.*$/prefix=\/opt\/sophon\/sophon-opencv-latest/g" /opt/sophon/sophon-opencv-latest/lib/pkgconfig/opencv4.pc
```
Finally, **Install bz2 libc6 libgcc dependent libraries** (this part requires selecting the corresponding installation package according to different operating systems, and will not be introduced uniformly here)
Then some configuration work:

Add library and executable paths:
```
sudo cp /opt/sophon/sophon-ffmpeg-latest/data/01_sophon-ffmpeg.conf /etc/ld.so.conf.d/
sudo cp /opt/sophon/sophon-opencv-latest/data/02_sophon-opencv.conf /etc/ld.so.conf.d/
sudo ldconfig
sudo cp /opt/sophon/sophon-ffmpeg-latest/data/sophon-ffmpeg-autoconf.sh /etc/profile.d/
sudo cp /opt/sophon/sophon-opencv-latest/data/sophon-opencv-autoconf.sh /etc/profile.d/
sudo cp /opt/sophon/sophon-sample-latest/data/sophon-sample-autoconf.sh /etc/profile.d/
source /etc/profile
```
For other platform machines, please refer to [libsophon installation tutorial](https://doc.sophgo.com/sdk-docs/v22.12.01/docs_latest_release/docs/sophon-mw/manual/html/1_install.html).
For more information about sophon-mw, please refer to "MULTIMEDIA User Manual.pdf" and "MULTIMEDIA Development Reference Manual.pdf".

### 5.3 Compile and install sophon-sail
The installation method is the same as [3.3 Compile and install sophon-sail](#33-Compile and install sophon-sail).