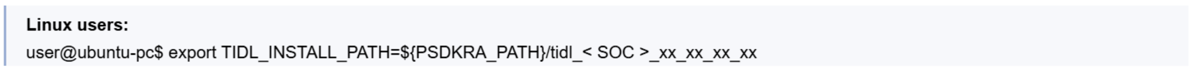
TIDL RT Build Instructions

Step 1: Export the required variables



Linux users:

```
user@ubuntu-pc$ export TIDL_INSTALL_PATH=${PSDKRA_PATH}/tidl_< SOC >_xx_xx_xx_xx
```

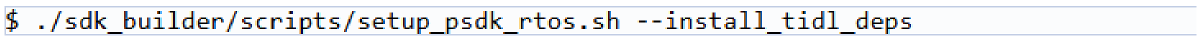
Set the TIDL_INSTALL_PATH with the TIDL root directory.
Example: ~/ti-processor-sdk-rtos-j721e-evm-09_00_01_01/c7x-mma-tidl

Step 2: Installing dependencies

All dependencies should be installed inside SDK path (~/ti-processor-sdk-rtos-j721e-evm-xx_xx_xx_xx$)

Install cmake if it not available in the system"sudo apt install cmake"

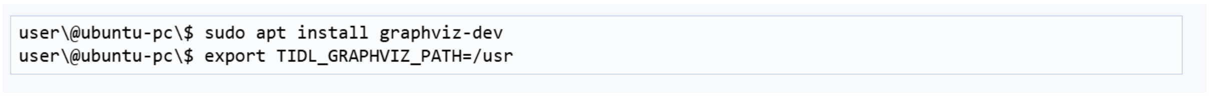
These dependencies can also be automatically downloaded & built by the following script in your SDK:



```
$ ./sdk_builder/scripts/setup_psdk_rtos.sh --install_tidl_deps
```

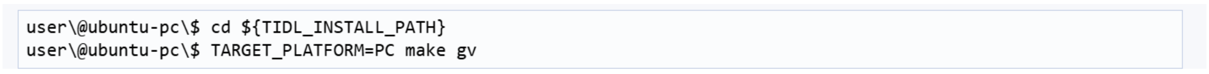
Else, you can install them separately:

1) **GraphViz tool:**
   To build tidlModelGraphviz tool, install graphviz-dev package and export the variable TIDL_GRAPHVIZ_PATH.



```
user\@ubuntu-pc\$ sudo apt install graphviz-dev
user\@ubuntu-pc\$ export TIDL_GRAPHVIZ_PATH=/usr
```

   Build the tidlModelGraphviz tool by running make in TIDL_INSTALL_PATH.



```
user\@ubuntu-pc\$ cd ${TIDL_INSTALL_PATH}
user\@ubuntu-pc\$ TARGET_PLATFORM=PC make gv
```

2) **Google protobuf:**
   Use the wget command to download the file.
   wget
   https://github.com/protocolbuffers/protobuf/releases/download/v3.11.3/protobuf-cpp-3.11.3.tar.gz

   Once the file is downloaded, extract it using tar:
   tar -xvzf protobuf-cpp-3.11.3.tar.gz

3) **Google Flatbuffers:**

Use the wget command to download the file.
wget https://github.com/google/flatbuffers/archive/v1.12.0.zip

Once its downloaded, extraxt it using tar:
tar -xvzf flatbuffers-1.12.0.tar.gz

4) **OpenCV:**
Use the wget command to download the file.
wget https://github.com/opencv/opencv/archive/4.1.0.zip

Once its downloaded, extraxt it using tar:
tar -xvzf opencv-4.1.0.tar.gz

Step 3: Building dependencies

Build OpenCV from source:

Use below CMake options to in "opencv-4.1.0/cmake" folder. And run "make" from same folder

```
cmake -DBUILD_opencv_highgui:BOOL="1" -DBUILD_opencv_videoio:BOOL="0" -DWITH_IPP:BOOL="0" -DWITH_WEBP:BOOL="1" -DWITH_OPENEXR:BOOL="1" -
        DWITH_IPP_A:BOOL="0" -DBUILD_WITH_DYNAMIC_IPP:BOOL="0" -DBUILD_opencv_cudacodec:BOOL="0" -DBUILD_PNG:BOOL="1" -
        DBUILD_opencv_cudaobjdetect:BOOL="0" -DBUILD_ZLIB:BOOL="1" -DBUILD_TESTS:BOOL="0" -DWITH_CUDA:BOOL="0" -
        DBUILD_opencv_cudafeatures2d:BOOL="0" -DBUILD_opencv_cudaoptflow:BOOL="0" -DBUILD_opencv_cudawarping:BOOL="0" -
        DINSTALL_TESTS:BOOL="0" -DBUILD_TIFF:BOOL="1" -DBUILD_JPEG:BOOL="1" -DBUILD_opencv_cudaarithm:BOOL="0" -
        DBUILD_PERF_TESTS:BOOL="0" -DBUILD_opencv_cudalegacy:BOOL="0" -DBUILD_opencv_cudaimgproc:BOOL="0" -
        DBUILD_opencv_cudastereo:BOOL="0" -DBUILD_opencv_cudafilters:BOOL="0" -DBUILD_opencv_cudabgsegm:BOOL="0" -
        DBUILD_SHARED_LIBS:BOOL="0" -DWITH_ITT=OFF ../
```

Build protobuf from source:

Run below Configure command in "protobuf-3.11.3" folder and rum "make" from the same folder

```
./configure CXXFLAGS=-fPIC --enable-shared=no LDFLAGS="-static"
```

Flatbuffer:

Run the following steps to build flatbuffers:

```
cd flatbuffers-1.12.0
cmake -G "Unix Makefiles" -DCMAKE_POSITION_INDEPENDENT_CODE=ON
make
```

Tensorflow repo:

This dependency is needed for Tensorflow-lite runtime specific builds.

```
cd ${PSDKRA_PATH}
git clone --depth 1 --single-branch -b tidl-j7 https://github.com/TexasInstruments/tensorflow.git
```

ONNX Repo:

This dependency is needed for ONNX runtime specific builds.

```
cd ${PSDKRA_PATH}
git clone --depth 1 --single-branch -b tidl-j7 https://github.com/TexasInstruments/onnxruntime.git
```

TVM Repo:

This dependency is needed for TVM/Neo-AI-DLR specific builds.

```
cd ${PSDKRA_PATH}
git clone --single-branch -b tidl-j7 https://github.com/TexasInstruments/tvm
cd tvm
git submodule init
git submodule update --init --recursive
```

Step 4: Building TIDL PC Tools:

Setting the environment variables:

Export the following variables needed for build:

```
export SOC=< SOC > #It should be set to one of (j721e, j721s2, j784s4, j722s, am62a, j742s2). Refer to the $SOC variable in ./sdk_builder/build_flags.mak
```

The following commands will build the host emulation (x86) tools.

```
$ cd sdk_builder
$ make tidl_pc_tools -j
```

Step 5: Build commands to run TIDL-RT:

Run “make TARGET_PLATFORM=PC” from ${TIDL_INSTALL_PATH} folder to build PC tools

```
$ cd ${TIDL_INSTALL_PATH}
$ make TARGET_PLATFORM=PC
```

This step will generate all the binaries for PC

- tidl_model_import.out in "ti_dl/utils/tidlModelImport/out"
- PC_dsp_test_dl_algo.out in "ti_dl/test"

Run “make” from ${TIDL_INSTALL_PATH} folder to build the test bench for target

```
$ cd ${TIDL_INSTALL_PATH}
$ make
```

- This Step will generate binary (./TI_DEVICE_dsp_test_dl_algo.out) in "ti_dl/test"

Note: All the commands can be found in the document provided in the sdk. It can be found in the following path: ~/ti-processor-sdk-rtos-j721e-evm-09_00_01_01/c7x-mma-tidl/ti_dl/docs/user_guide_html/md_tidl_overview.html

Reference Document for building x86 host emulation tools (Step 4): https://software-dl.ti.com/jacinto7/esd/processor-sdk-rtos-j722s/10_01_00_04/exports/docs/c7x-mma-tidl/ti_dl/docs/user_guide_html/md_tidl_build_instruction.html

Step 6: Running Custom Test App

To build a custom App, say, hello world.c,

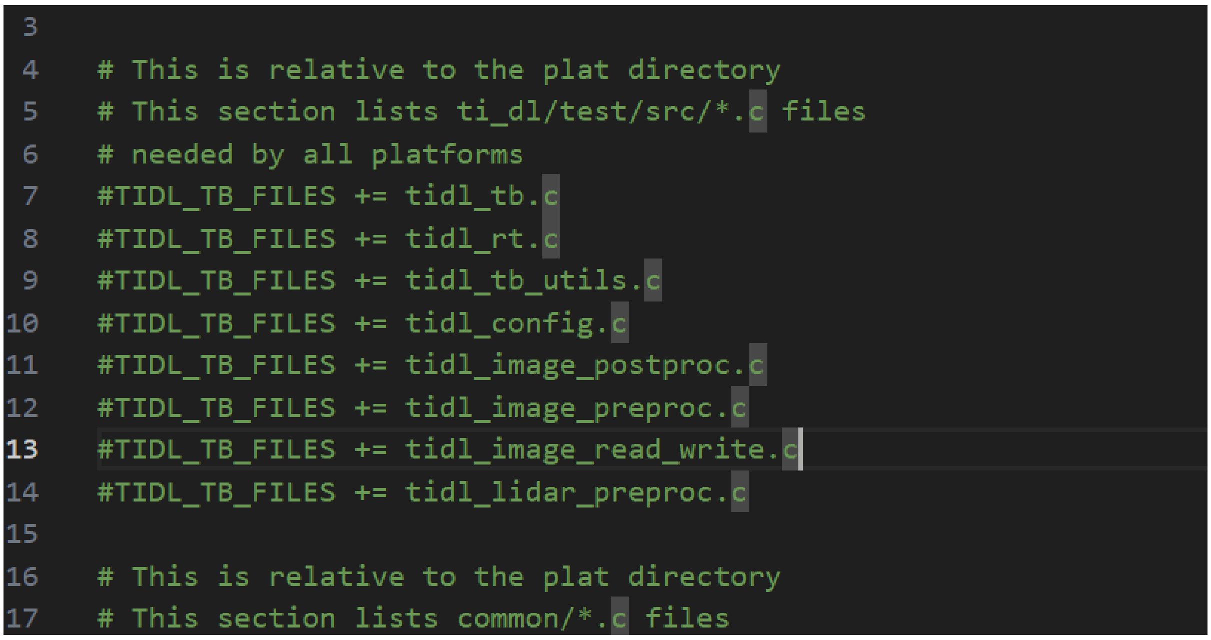
1) **Navigate to the Directory**
   Go to the following directory where the source files are located:

   ~/ti-processor-sdk-rtos-j721e-evm-09_00_01_01/c7x-mma-tidl/ti_dl/test/src/pc_linux/

2) **Study Existing Files**
   Review the main.c file and the associated makefiles in the directory. These files are responsible for generating the PC_dsp_test_dl_algo.out file.
3) **Modify the main.c File**
   Replace the existing code in the main.c file with the custom hello_world.c code.
   Delete the platform_defines.h file from the directory.

4) **Remove Unnecessary Files**
   Delete all other .c and .h files from the src folder except for the main.c file with your custom code.

5) **Update the Makefile**
   Open the concerto_common.mak file.
   Comment out the section that lists ti_dl/test/src/*.c files needed by all platforms.



```

# This is relative to the plat directory
# This section lists ti_dl/test/src/*.c files
# needed by all platforms
#TIDL_TB_FILES += tidl_tb.c
#TIDL_TB_FILES += tidl_rt.c
#TIDL_TB_FILES += tidl_tb_utils.c
#TIDL_TB_FILES += tidl_config.c
#TIDL_TB_FILES += tidl_image_postproc.c
#TIDL_TB_FILES += tidl_image_preproc.c
#TIDL_TB_FILES += tidl_image_read_write.c
#TIDL_TB_FILES += tidl_lidar_preproc.c

# This is relative to the plat directory
# This section lists common/*.c files
```

6) **Backup Original Files**
   Ensure you keep a backup of the original src folder in case you need to revert changes.

7) **Build the Custom Application**
   Run the following command to build the application:
   make TARGET_BUILD=debug TARGET_PLATFORM=PC

8) **Execute the Generated File**
   Once the build is complete, run the generated output file using the following command:
   ./PC_dsp_test_dl_algo.out