**Qualcomm® QCS610 SOC Open Kit smarttraffic-demo2.0 Developer documentation**

1. **Introduce**

　　This project is based on an open source [EasyPR](https://gitee.com/easypr/EasyPR) and Qualcomm Neural processing SDK for AI. Using the Qualcomm® QCS610 SOC Open Kit and IMX415 camera module. Then, its powerful computing ability will been show by the license plate recognition process is completed that combine with Qualcomm Neural processing SDK for AI, OpenCV, SVM and Gstreamer. The EasyPR license plate recognition engine is transplanted to the Qualcomm® QCS610 SOC development board to display the rich application scenes of Qualcomm® QCS610 SOC. At the same time, UI is developed UI by GTK to display the progress of license plate recognition.

The project was built in x86 host with across complier tool and has been tested in Qualcomm® QCS610 SOC device.

Qualcomm® QCS610 SOC Development board

****

1. **Materials and Tools used for the project**

**2.1 Other Hardware materials**

Except for the development board, the following hardware materials are also needed:

**2.1.1 Type-C USB line**

Using the USB line to develop on Qualcomm® QCS610 SOC development board.



**2.1.2 Charger**

Direct power supply for Qualcomm® QCS610 SOC development board.



**2.1.3 DP-Line**

Using the universal DP line to connect LED displayer to Qualcomm® QCS610 SOC development board.



**2.1.4 LED-Displayer**

Using a LED Displayer to display the smarttraffic-demo2.0 interface from Qualcomm® QCS610 SOC development board.



1. **Environment configuration**
   1. **Qualcomm Neural processing SDK for AI**
2. Download the Neural Processing SDK from here:

<https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk>

1. Git clone this project link

<https://github.com/ThunderSoft-XA/C610-smarttraffic-demo2.0/>

1. Move the Neural Processing SDK contents to <C610-smarttraffic-demo2.0/snpe/include/zdl>

**3.2 GTK-3.0+**

The demo`s UI interface based on GTK with dependence library.For example,cario,glib,pango,etc.So,you need install it in progress of host development if you want to program it again,as follow:

1. install gcc/g++/gdb/make etc basic compile tools :

sudo apt-get install build-essential

2. install GTK+3.0：

sudo apt-get install libgtk-3-dev

3. install pkg-config：

sudo apt-get install pkg-config

4. install help files：

sudo apt-get install devhelp

5. check version of pkg-config/gtk+:

pkg-config -–version

pkg-config -–modversion gtk+-3.0

1. **Compile**

　　The compilation of the whole project is based on the Yocto compilation tool, so you need to write some .bb and .conf files according to the specification. The traffic\_0.1.bb example is as follows:

inherit cmake

DESCRIPTION = "smarttraffic-demo2.0"

LICENSE = "BSD"

SECTION = "smarttraffic-demo2.0"

LIC\_FILES\_CHKSUM="file://${COMMON\_LICENSE\_DIR}/${LICENSE};md5=3775480a712fc46a69647678acb234cb"

# Dependencies.

DEPENDS := "opencv gtk+3 sqlite3 glib-2.0 dbus-glib glib-2.0-native"

DEPENDS += "zlib fontconfig cairo gst-plugins-base gconf libpng pango"

DEPENDS += "gstreamer1.0"

DEPENDS += "gstreamer1.0-plugins-base"

DEPENDS += "gstreamer1.0-plugins-qti-oss-mlmeta"

DEPENDS += "gstreamer1.0-plugins-qti-oss-tools"

DEPENDS += "gstreamer1.0-rtsp-server"

EXTRA\_OECONF += " --with-glib"

CPPFLAGS += " -I${STAGING\_INCDIR}/glib-2.0"

CPPFLAGS += " -I${STAGING\_LIBDIR}/glib-2.0/include"

CPPFLAGS += "-include glib.h"

CPPFLAGS += "-include glibconfig.h"

LDFLAGS += " -lglib-2.0"

FILESPATH =+ "${WORKSPACE}/video\_ai/camera/bin/:"

SRC\_URI = "file://camera/"

INSANE\_SKIP\_${PN}-dev += "ldflags dev-elf dev-deps"

PACKAGES = "${PN}-dbg ${PN} ${PN}-dev"

S = "/home/turbox/wuqx0806/cs-610/apps\_proc/src/video\_ai/camera/"

# Install directries.

INSTALL\_INCDIR := "${includedir}"

INSTALL\_BINDIR := "${bindir}"

INSTALL\_LIBDIR := "${libdir}"

EXTRA\_OECMAKE += ""

FILES\_${PN} += "${INSTALL\_BINDIR}"

FILES\_${PN} += "${INSTALL\_LIBDIR}"

SOLIBS = ".so\*"

FILES\_SOLIBSDEV = ""

　　Please refer to [the official Manual of Yocto](https://www.yoctoproject.org) for how to add layers, write layer.conf. Then, execute the command as follows:

bitbake smarttraffic-demo2.0

　　You will get an executable bin file named smarttraffic-demo2.0.Move it to the root of the source code. Next, push the whole project code to Qualcomm® QCS610 SOC device`s directory /data/.

$ adb root && adb disable-verity && adb reboot

$ adb root &&adb remount && adb shell mount -o remount,rw /

$ adb push xxx/xxx/sourcepath /data

$ adb shell

$ cd data/smarttraffic-demo2.0

1. **Configure Weston and Usage**

**4.1 Configure Weston**

To Download *Turbox-C610-aarch64\_AI\_Demo\_Firmware.tgz*, go to: TBU

Push the ***firmware package*** to target device`s **/data/** directory.

$ adb root

$ adb disable-verity && adb reboot

$ adb root && adb shell mount -o remount,rw /

$ adb push Turbox-C610-aarch64\_AI\_Demo\_Firmware.tgz /data/

$ adb shell

$ tar -zxvf /data/ Turbox-C610-aarch64\_AI\_Demo\_Firmware.tgz -C /data/

Unpack *Turbox-C610-aarch64\_Weston\_DP\_Firmware.tgz* under root directory to enable weston output to DP.

$ tar -zxvf /data/ Turbox-C610-aarch64\_AI\_Demo\_Firmware/aarch64-weston-dp.tgz -C /

**4.2 Usage**

This project only provides a simple command line interface.

run smarttraffic-demo2.0：

Run the **smarttraffic-demo2.0** app. The demo video will play on the DP screen.

$ cd /data/<source root dir>

$ ./weston\_dp\_client smarttraffic-demo2.0

**weston\_dp\_client :**

#!/bin/sh

echo "=====Configure Weston environment====="

mount -o remount,rw /

killall weston

#mkdir /usr/bin/weston\_socket

export XDG\_RUNTIME\_DIR=/dev/socket/weston

#mkdir --parents XDG\_RUNTIME\_DIR

chmod 0700 $XDG\_RUNTIME\_DIR

cd /usr/bin

./weston --tty=1 --device=msm\_drm --idle-time=0 &

sleep 2

echo "=====Show the Weston flower====="

sleep 1

sh weston-flower &

sleep 3

if [[ $1 = "smarttraffic-demo2.0" ]]; then

cd /data/camera/

echo "=====Show the Smart Traffic Demo 2.0====="

sleep 1

./smarttraffic-demo2.0 &

sleep 1800

killall smarttraffic-demo2.0

elif [[ $1 = "decode" ]]; then

echo "=====Show the Multi Decode====="

sleep 1

multi-decoder 6 /data/Turbox-C610-aarch64\_AI\_Demo\_Firmware/config.ini &

sleep 20

killall multi-decoder

else

echo "=====Missing parameter====="

sleep 2

fi

echo "=====Turn off Weston display====="

killall weston