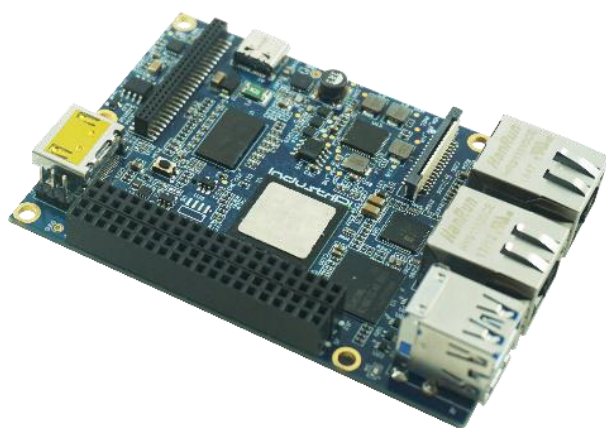


IndustriPi

软件使用手册 V2.0



发布说明

| 日期 | 版本 | 发布说明 |
|------------|------|------------|
| 2018-10-30 | V1.0 | 初始版本 |
| 2019-06-13 | V2.0 | 修改 1.7 章节 |
| | | 修改第 2 章节 |
| | | 添加 TIDL 框架 |

免责声明

本文中的信息，包括参考的 URL 地址，如有变更，恕不另行通知。文档“按现状”提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档不负任何责任，包括使用本文档内信息产生的侵犯任何专利权行为的责任。本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

版权公告

版权归©北京匠牛科技有限公司。保留所有权利。

目录

| | |
|----------------------|-----------|
| 第 1 章 快速入门 | 1 |
| 1.1 开机登录 | 1 |
| 1.2 网口测试 | 2 |
| 1.2.1 RGMII 网口 | 2 |
| 1.2.2 PRU-MII 网口 | 2 |
| 1.3 USB 测试 | 3 |
| 1.4 Micro SD 测试 | 4 |
| 1.5 HDMI 接口测试 | 5 |
| 1.6 LED 灯测试 | 6 |
| 1.7 查询系统信息 | 6 |
| 第 2 章 SDK 详解 | 9 |
| 2.1 SDK 安装 | 9 |
| 2.2 SDK 组件介绍 | 10 |
| 2.3 安装交叉编译工具链 | 11 |
| 2.4 交叉编译工具链特性介绍 | 11 |
| 2.5 构建 MLO 和 u-boot | 12 |
| 2.6 构建 Linux 内核 | 12 |
| 2.7 固件更新 | 14 |
| 2.7.1 更新 SD 卡 | 14 |
| 2.7.2 更新 emmc | 15 |
| 2.8 制作 SD 启动卡 | 15 |
| 2.9 烧写 emmc | 16 |
| 第 3 章 图形显示框架 | 17 |
| 3.1 QT5 图形框架 | 18 |
| 第 4 章 多媒体框架 | 19 |
| 第 5 章 TIDL 框架 | 21 |
| 5.1 TIDL 案例 | 22 |

第 1 章 快速入门

1.1 开机登录

1. 将开发板 UART3 串口(J1)通过 USB 转 TTL 串口线连接到 PC 机 USB 接口。

对应线序：

| 描述 | J1 接口 | USB 转串口 |
|----|-------|---------|
| 接收 | R | 绿色 |
| 发送 | T | 白色 |
| 地 | G | 黑色 |

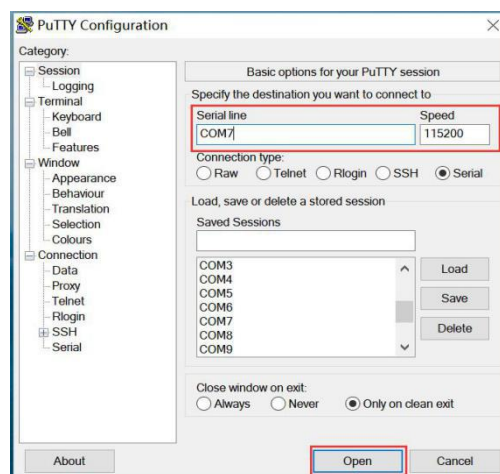
2. 查看 USB 转串口设备端口

点击桌面 " 此电脑 " 图标，右击选择 " 管理 "，点击设备管理器查看 USB 转串口设备端口。



3. 打开 putty 软件，按照下图进行参数配置：

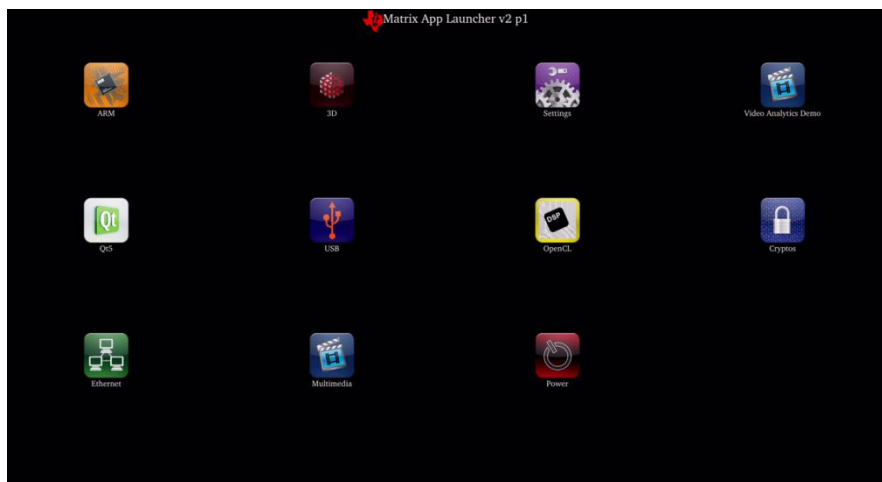
Connection type 选择 Serial，Serial line 选择相应 COM 口，按照第三步骤查找对应端口，Speed 选择 115200,其他选择默认参数，最后点击 Open 连接。



4. HDMI 线连接到板卡 HDMI 输出接口(P6)。
5. 接通电源，将配件中 Type-c 线连接到 J11 接口。

UART3 串口会打印 U-boot、内核和文件系统的调试信息。

6. 系统启动完成后，HDMI 输出 TI Processor SDK linux 自带的 Matrix 用户界面，如下图所示。



7. 通过 putty 软件进入 Linux 登录界面，输入账号 root，密码为空，回车即可完成登录。



注意：工业派自带嵌入式 Linux 系统。推荐使用 5v/2A 以上电源接头。先连接 HDMI 接口，再上电。
支持 HDMI 转 HDMI 线，HDMI 转 DVI 线，不支持 HDMI 转 VGA 线。

1.2 网口测试

工业派支持 J3 RGMII(10M/100M/1000M)千兆网和 P1 PRU-MII(10M/100M)百兆网两个网口。

1.2.1 RGMII 网口

- ◆ 查看 eth0 IP 地址：

```
root@am57xx-evm:~# ifconfig eth0
```

- ◆ ping 匠牛社区官网

```
root@am57xx-evm:~# ping www.jiang-niu.com
```

1.2.2 PRU-MII 网口

- ◆ 关闭 eth0：

```
root@am57xx-evm:~# ifconfig eth0 down
```

◆ 查看 eth1 IP 地址:

```
root@am57xx-evm:~# ifconfig eth1
```

◆ ping 匠牛社区官网

```
root@am57xx-evm:~# ping www.jiang-niu.com
```

1.3 USB 测试

J2 为 USB 叠层接口，上边为 USB2.0 接口，下边为 USB3.0 接口。

1. USB 热插拔测试

将 2.0 U 盘连接到 USB2.0 接口，热插拔信息如下图 1-3-1 所示:

```
[ 36.787516] usb 3-1: new high-speed USB device number 2 using xhci-hcd
[ 37.115454] usb-storage 3-1:1.0: USB Mass Storage device detected
[ 37.132016] scsi host1: usb-storage 3-1:1.0
[ 37.136600] usbcore: registered new interface driver usb-storage
[ 38.138403] scsi 1:0:0:0: Direct-Access    Teclast  CoolFlash      8.01 PQ: 0 ANSI: 2
[ 38.141007] scsi 1:0:0:1: CD-ROM          Generic  Autorun Disk    8.00 PQ: 0 ANSI: 2
[ 38.153177] sd 1:0:0:0: [sda] 7638398 512-byte logical blocks: (3.91 GB/3.64 GiB)
[ 38.153403] sd 1:0:0:0: [sda] Write Protect is off
[ 38.153411] sd 1:0:0:0: [sda] Mode Sense: 03 00 00 00
[ 38.153661] sd 1:0:0:0: [sda] No Caching mode page found
[ 38.153667] sd 1:0:0:0: [sda] Assuming drive cache: write through
[ 38.197077] sr 1:0:0:1: [sr0] scsi3-mmc drive: 0x/0x caddy
[ 38.197081] cdrom: Uniform CD-ROM driver Revision: 3.20
[ 38.197545] sda: sda4
[ 38.215175] sr 1:0:0:1: Attached scsi CD-ROM sr0
[ 38.216627] sd 1:0:0:0: [sda] Attached SCSI removable disk
[ 39.012430] FAT-fs (sda4): Volume was not properly unmounted. Some data may be corrupt.
```

图 1-3-1

将 3.0 U 盘连接到 USB3.0 接口，热插拔信息如下图 1-3-2 所示:

```
[ 110.457898] usb 2-1: new SuperSpeed USB device number 2 using xhci-hcd
[ 110.496333] usb-storage 2-1:1.0: USB Mass Storage device detected
[ 110.512863] scsi host2: usb-storage 2-1:1.0
[ 111.518132] scsi 2:0:0:0: Direct-Access    Kingston DataTraveler 3.0 PQ: 0 ANSI: 6
[ 111.520767] sd 2:0:0:0: [sdb] 30218842 512-byte logical blocks: (15.5 GB/14.4 GiB)
[ 111.520994] sd 2:0:0:0: [sdb] Write Protect is off
[ 111.521002] sd 2:0:0:0: [sdb] Mode Sense: 4f 00 00 00
[ 111.521228] sd 2:0:0:0: [sdb] Write cache: disabled, read cache: enabled, doesn't support DPO
[ 111.524666] sdb: sdb1
[ 111.564800] sd 2:0:0:0: [sdb] Attached SCSI removable disk
[ 111.797341] FAT-fs (sdb1): Volume was not properly unmounted. Some data may be corrupt. Please
```

图 1-3-2

从图 1-3-1 可知 USB 3-1 成功挂载了一个 3.91 GB 的 2.0 U 盘，挂载名是 sda，该 U 盘只有一个分区 sda4，

从图 1-3-2 可知 USB 2-1 成功挂载了一个 15.5 GB 的 3.0 U 盘，挂载名是 sdb，该 U 盘只有一个分区 sdb1。

注意：U 盘挂载名请以实际操作为准，U 盘插拔顺序不同，挂载名称也会不同。

2. 查看 U 盘挂载信息

```
root@am57xx-evm:~# df
```

```

root@am57xx-evm:~# df

```

| Filesystem | 1K-blocks | Used | Available | Use% | Mounted on |
|----------------|-----------|---------|-----------|------|----------------------|
| /dev/root | 7289224 | 2783744 | 4112164 | 40% | / |
| devtmpfs | 840044 | 4 | 840040 | 0% | /dev |
| tmpfs | 934420 | 8 | 934412 | 0% | /dev/shm |
| tmpfs | 934420 | 9744 | 924676 | 1% | /run |
| tmpfs | 934420 | 0 | 934420 | 0% | /sys/fs/cgroup |
| tmpfs | 934420 | 4 | 934416 | 0% | /tmp |
| tmpfs | 51200 | 12 | 51188 | 0% | /var/volatile |
| tmpfs | 16384 | 0 | 16384 | 0% | /media/ram |
| /dev/mmcblk0p1 | 77619 | 452 | 77167 | 1% | /run/media/mmcblk0p1 |
| tmpfs | 186884 | 0 | 186884 | 0% | /run/user/0 |
| /dev/sda4 | 3810972 | 756 | 3810216 | 0% | /run/media/sda4 |
| /dev/sdb1 | 15100688 | 170064 | 14930624 | 1% | /run/media/sdb1 |

从上图可知：

2.0 U 盘挂载目录是/run/media/sda4

3.0 U 盘挂载目录是/run/media/sdb1

3. 拷贝数据

拷贝内核日志到 2.0 U 盘

```

root@am57xx-evm:~# dmesg > kernel_log
root@am57xx-evm:~# cp kernel_log /run/media/sda1/
root@am57xx-evm:~# sync

```

拷贝数据到 IndustriPi 家目录

```
root@am57xx-evm:~# cp -r /run/media/sda4/JiangNiu-demo /home/root
```

4. 卸载测试

```
root@am57xx-evm:~# umount /run/media/sda4
```

```
root@am57xx-evm:~# umount /run/media/sdb1
```

1.4 Micro SD 测试

P2 为 Micro SD 卡座，支持 Class 4，Class 10 Micro SD 卡。

1. Micro SD 卡热插拔测试

将 Micro SD 卡连接到 Micro SD 卡座，热插拔信息如下图 1-4-1 所示：

```

root@am57xx-evm:~# dmesg
[ 33.686787] mmc0: host does not support reading read-only switch, assuming write-enable
[ 33.698846] mmc0: new high speed SDHC card at address aaaa
[ 33.714972] mmcblk1: mmc0:aaaa SL08G 7.40 GiB
[ 33.730899] mmcblk1: p1
[ 34.061472] EXT4-fs (mmcblk1p1): recovery complete
[ 34.066301] EXT4-fs (mmcblk1p1): mounted filesystem with ordered data mode. Opts: (null)

```

从上图可知 mmc0 成功挂载了一个 7.4GB 的 Micro SD 卡，挂载名是 mmcblk1，

该 SD 卡只有一个分区 mmcblk1p1。

注意：Micro SD 卡挂载名请以实际操作为准，板卡启动方式不同，挂载名称也会不同。

2. 查看 Micro SD 卡挂载信息

```
root@am57xx-evm:~# df
```

```
root@am57xx-evm:~# df
```

| Filesystem | 1K-blocks | Used | Available | Use% | Mounted on |
|----------------|-----------|---------|-----------|------|----------------------|
| /dev/root | 7289224 | 2783792 | 4112116 | 40% | / |
| devtmpfs | 840044 | 4 | 840040 | 0% | /dev |
| tmpfs | 934420 | 8 | 934412 | 0% | /dev/shm |
| tmpfs | 934420 | 9716 | 924704 | 1% | /run |
| tmpfs | 934420 | 0 | 934420 | 0% | /sys/fs/cgroup |
| tmpfs | 934420 | 4 | 934416 | 0% | /tmp |
| tmpfs | 51200 | 12 | 51188 | 0% | /var/volatile |
| tmpfs | 16384 | 0 | 16384 | 0% | /media/ram |
| /dev/mmcblk0p1 | 77619 | 452 | 77167 | 1% | /run/media/mmcblk0p1 |
| tmpfs | 186884 | 0 | 186884 | 0% | /run/user/0 |
| /dev/mmcblk1p1 | 65876 | 1820 | 58998 | 3% | /run/media/mmcblk1p1 |

从上图可知：

Micro SD 卡挂载目录是/run/media/mmcblk1p1

3. 拷贝数据

拷贝内核日志到 Micro SD 卡

```
root@am57xx-evm:~# dmesg > kernel_log
root@am57xx-evm:~# cp kernel_log /run/media/mmcblk1p1/
root@am57xx-evm:~# sync
```

从 Micro SD 卡拷贝数据到 IndustriPi 家目录

```
root@am57xx-evm:~# cp -r /run/media/mmcblk1p1/JiangNiu-demo /home/root
```

4. 卸载测试

```
root@am57xx-evm:~# umount /run/media/mmcblk1p1
```

1.5 HDMI 接口测试

使用 HDMI 线连接开发板的 HDMI 输出接口到显示器或带有 HDMI 接口的 LCD 屏，系统启动完成，外接显示器将显示 TI Processor SDK linux 自带的 Matrix 用户界面。IndustriPi 支持 HDMI 1.4a, HDCP 1.4, DVI 1.0 协议。

1. 关闭 Matrix 界面：

```
root@am57xx-evm:~# /etc/init.d/matrix-gui-2.0 stop
```

2. 停止 weston 显示服务：

```
root@am57xx-evm:~# /etc/init.d/weston stop
```


3. 启动 weston 显示服务:

```
root@am57xx-evm:~# /etc/init.d/weston start
```

4. 打开 Matrix 界面:

```
root@am57xx-evm:~# /etc/init.d/matrix-gui-2.0 start
```

5. HDMI 音频测试

```
root@am57xx-evm:~# aplay -Dhw:1,0 /usr/share/sounds/alsa/Front_Center.wav
```

注意:

1. Matrix 界面依赖 weston 显示服务, 因此打开 Matrix 界面之前, 请确保 weston 显示服务已经启动。
2. HDMI 音频测试时, 请将 HDMI 接口连接到 HDMI 电视机(支持音频输出)。

1.6 LED 灯测试

IndustriPi 有 1 组绿色 LED 灯 D4, 位置在 USB 接口侧边。

1. 点亮 LED

```
root@am57xx-evm:~# echo 1 > /sys/class/leds/status_usr0/brightness
```

2. 关闭 LED

```
root@am57xx-evm:~# echo 0 > /sys/class/leds/status_usr0/brightness
```

1.7 查询系统信息

1. 查看主机名称

```
root@am57xx-evm:~# hostname
```

```
root@am57xx-evm:~# hostname
am57xx-evm
root@am57xx-evm:~#
```

主机名称记录在/etc/hostname 文件, 可以直接修改文件内容改变主机名称

2. 查看 Linux 内核版本

```
root@am57xx-evm:~# uname -a
```

```
root@am57xx-evm:~# uname -a
Linux am57xx-evm 4.9.69-g9ce43c71ae #19 SMP PREEMPT Tue Jan 8 11:32:11 CST 2019 armv7l GNU/Linux
root@am57xx-evm:~#
```

注意: 上图是 SDK4.3 内核版本, 具体情况以实际为准。

3. 查看 Linux 内核启动参数

```
root@am57xx-evm:~# cat /proc/cmdline
```

```
root@am57xx-evm:~# cat /proc/cmdline
console=ttyO2,115200n8 root=PARTUUID=6ad18f5f-02 rw rootfstype=ext4 rootwait
```

4. 查看 Linux 系统环境变量

```
root@am57xx-evm:~# env
```

```
root@am57xx-evm:~# env
XDG_SESSION_ID=c1
SSL_CERT_FILE=/etc/ssl/certs/ca-certificates.crt
WAYLAND_DISPLAY=wayland-0
SHELL=/bin/sh
TERM=xterm
QT_QPA_EGLFS_INTEGRATION=none
QT_QPA_EGLFS_KMS_CONFIG=/etc/qt5/eglfs_kms_cfg.json
HUSHLOGIN=FALSE
USER=root
WS_CALUDEV_FILE=/etc/udev/rules.d/ws-calibrate.rules
MAIL=/var/spool/mail/root
PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/sbin:/usr/sbin:/sbin
PWD=/home/root
EDITOR=vi
PS1=\u@\h:\w\$
SHLVL=1
HOME=/home/root
XDG_CONFIG_HOME=/etc/
LOGNAME=root
XDG_RUNTIME_DIR=/tmp/0-runtime-dir
_=/usr/bin/env
root@am57xx-evm:~# |
```

5. 查看中断信息

```
root@am57xx-evm:~# cat /proc/interrupts
```

```
root@am57xx-evm:~# cat /proc/interrupts
          CPU0           CPU1
16:         0             0      CBAR 32 Level    gp_timer
19:    188017    188955      GIC 27 Edge     arch_timer
22:         0             0      CBAR  4 Level    l3-dbg-irq
23:         0             0    WUGEN 10 Level    l3-app-irq
25:         1             0      CBAR 121 Level  talert
27:        106             0      CBAR  8 Level    omap-dma-engine
30:         810             0      CBAR 361 Level  43300000.edma_ccint
32:         0             0      CBAR 359 Level  43300000.edma_ccerrint
35:         1             0      CBAR 24 Level    4ae10000.gpio
36:         0             1  4ae10000.gpio  0 Level    palmas
68:         0             0      CBAR 25 Level    48055000.gpio
101:        0             0      CBAR 26 Level    48057000.gpio
134:        0             0      CBAR 27 Level    48059000.gpio
156:        0             0  48059000.gpio  21 Edge    palmas_usb_vbus
167:        0             0      CBAR 28 Level    4805b000.gpio
200:        0             0      CBAR 29 Level    4805d000.gpio
228:        0             0  4805d000.gpio  27 Edge    4809c000.mmc cd
233:        0             0      CBAR 30 Level    48051000.gpio
246:        0             0  48051000.gpio  12 Edge    tpd12s015 hpd
266:        0             0      CBAR 116 Level  48053000.gpio
299:       3194             0      CBAR 69 Level    48020000.serial
```

6. 查看 CPU 信息

```
root@am57xx-evm:~# cat /proc/cpuinfo
```

```
root@am57xx-evm:~# cat /proc/cpuinfo
processor       : 0
model name     : ARMv7 Processor rev 2 (v7l)
BogoMIPS      : 11.80
Features       : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
CPU implementer : 0x41
CPU architecture: 7
CPU variant    : 0x2
CPU part       : 0xc0f
CPU revision   : 2

Hardware       : Generic DRA72X (Flattened Device Tree)
Revision      : 0000
Serial        : 0601700c34f00322
```

7. 查看内存余量

```
root@am57xx-evm:~# free -h
```

```
root@am57xx-evm:~# free -h
              total        used        free      shared  buff/cache   available
Mem:           807M         178M         389M           74M          239M          541M
Swap:           0B           0B           0B
```

第 2 章 SDK 详解

2.1 SDK 安装

SDK 安装步骤如下所示：

1. 安装 Ubuntu 主机

使用 SDK 时，需要一台可用的 Linux 系统主机 PC 或者虚拟机。可用的 Linux 主机有很多，我们无法验证所有可能性，推荐使用 Ubuntu 64bit 作为开发主机。我们 SDK 发布时验证 ubuntu 64bit 的长期支持(LTS)版本(Ubuntu14.04 和 Ubuntu16.04)。详细安装请参考《Ubuntu 环境开发搭建》手册。如果您已有 Linux 主机，可跳过该步骤。

2. 下载 TI 官方 SDK 包

http://software-dl.ti.com/processor-sdk-linux/esd/AM57X/latest/index_FDS.html

PROCESSOR-SDK-LINUX-AM57X Product downloads

| Title | Description |
|---|---|
| AM57xx Linux SDK Essentials | |
| ti-processor-sdk-linux-am57xx-evm-04.03.00.05-Linux-x86-Install.bin | AM57xx EVM Linux SDK (64-bit Binary) |
| AM57xx Linux SDK Optional Addons | |
| ccs_setup_7.4.0.00015.exe | Code Composer Studio IDE for Windows Host |
| CCS7.4.0.00015_web_linux-x64.tar.gz | Code Composer Studio IDE for Linux Host |
| Download Pinmuxtool | AM57xx Pin Mux Configuration Utility |
| Wilink 8 Addon Package | Wilink 8 Addon Package |

注意：下载相应版本。目前支持 SDK4.3 和 SDK5.3 两个版本。

3. 下载 JN-IndustriPi 资料包

<http://www.jiang-niu.com/download.html>

4. Ubuntu 终端运行如下命令，安装 TI 官方 SDK 包

```
chmod 0777 ti-processor-sdk-linux-am57xx-evm-xx.xx.xx.xx-Linux-x86-Install.bin
```

```
./ti-processor-sdk-linux-am57xx-evm-xx.xx.xx.xx-Linux-x86-Install.bin
```

注意：安装路径最好选择家目录，即/home/xxx，xxx 为 Ubuntu PC 用户名。x 代表版本号，具体以实际为准。

5. Ubuntu 终端运行如下命令，安装 JN-IndustriPi 源码包

解压 JN_IndustriPi_x.x 资料包，进入源代码目录进行如下操作。

```
root@am57xx-evm:~# tar zxvf JN_IndustriPi_x.x
```

```
root@am57xx-evm:~#cp -r JN_IndustriPi_x.x/源代码/board-support_x.x.x.x ~/ti-processor-sdk-linux-rt-am57xx-evm-xx.
```

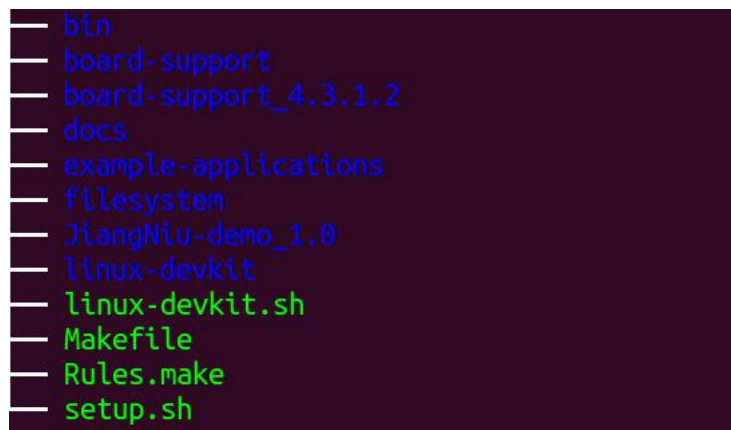
```
root@am57xx-evm:~#cp -r JN_IndustriPi_x.x/源代码/JiangNiu-demo_x.x.x.x ~/ti-processor-sdk-linux-rt-am57xx-evm-xx.
```

```
root@am57xx-evm:~#sync
```

注意：x 代表不同版本号,具体以实际为准。

2.2 SDK 组件介绍

SDK 组件如下图所示：



```
— bin
— board-support
— board-support_4.3.1.2
— docs
— example-applications
— filesystem
— JiangNiu-demo_1.0
— linux-devkit
— linux-devkit.sh
— Makefile
— Rules.make
— setup.sh
```

bin: Ubuntu PC 配置脚本

board-support: TI 官方 Linux 内核源码，U-boot 源码，扩展驱动源码以及预编译镜像

board-support_4.3.1.2: 工业派官方 Linux 内核源码，U-boot 源码

docs: TI 官方文档

example-applications: TI 官方实例

filesystem: 文件系统压缩包

linux-devkit: 交叉编译工具链和相关库文件，比如 Gstreamer 库，OpenCV 库，OpenCL 库，QT 库

(linux=devkit/sysroots/armv7ahf-neon-linux-gnueabi/usr/lib)

JiangNiu-demo_1.0: 工业派官方实例

2.3 安装交叉编译工具链

1. 打开.bashrc 文件

sudo vim ~/.bashrc

2. 添加如下命令到文件末尾，然后保存

```
export PATH=$PATH:~/ti-processor-sdk-linux-am57xx-evm-04.03.00.05/linux-devkit/
sysroots/x86_64-arago-linux/usr/bin
```

3. Ubuntu PC 运行如下命令，使 PATH 环境变量生效

source ~/.bashrc

4. Ubuntu PC 运行如下命令，测试交叉编译工具链是否安装成功

arm-linux-gnueabihf-gcc -v

打印信息如下图 1-3 所示，表示交叉编译工具链安装成功

```
quan@ubuntu:~$ arm-linux-gnueabihf-gcc -v
Using built-in specs.
COLLECT_GCC=arm-linux-gnueabihf-gcc
COLLECT_LTO_WRAPPER=/home/quantum/ti-processor-sdk-linux-am57xx-evm-04.03.00.05/linux-devkit/sysroots/x86_64-arago-linux/usr/bin/../libexec/gcc/arm-linux-gnueabihf/6.2.1/collect2.LTO_WRAPPER-arago-linux-gnueabihf
Target: arm-linux-gnueabihf
Configured with: /home/quantum/ti-processor-sdk-linux-am57xx-evm-04.03.00.05/linux-devkit/sysroots/x86_64-arago-linux/usr/bin/../libexec/gcc/arm-linux-gnueabihf/6.2.1/collect2.LTO_WRAPPER-arago-linux-gnueabihf
Thread model: posix
gcc version 6.2.1 20161016 (Linaro GCC 6.2-2016.11)
```

2.4 交叉编译工具链特性介绍

交叉编译工具链特性如下图所示：

| | |
|--------|---|
| C 语言特性 | 支持 C89(ISO/IEC 9899:1990) 支持 C99(ISO/IEC 9899:1999) 不支持 C11 (ISO/IEC 9899:2011) |
| C++特性 | 支持 C++98(ISO/IEC 14882:1998) 支持 C++03(ISO/IEC 14882:2003) 不支持 C++11(ISO/IEC 14882:2011) |

2.5 构建 MLO 和 u-boot

1. 清除源码

```
make CROSS_COMPILE=arm-linux-gnueabi- distclean
```

2. 编译 U-boot

同时编译 u-boot 和 SPL，执行如下命令去编译 u-boot

```
cd ~/ti-processor-sdk-linux-am57xx-evm-xx
```

```
cd board-support_x.x.x.x/u-boot-2018.01+gitAUTOINC+313dcd69c2-g825ac6e1ac
```

```
./build.sh
```

编译时间约 2 分钟，编译成功后，生成 MLO，u-boot.img 镜像文件。

注意：

配置文件目录：include/configs

引脚复用目录：board/ti/dra7xx/

| 版本 | 配置文件 |
|--------|--------------------------|
| SDK4.3 | JN-industrPi_mux_data.h |
| SDK4.3 | JN-IndustriPi_mux_data.h |

2.6 构建 Linux 内核

1. 清除源码

在编译 Linux 内核之前，确保内核源代码是干净的并没有以前构建时遗留文件

```
make CROSS_COMPILE=arm-linux-gnueabi- distclean
```

2. 编译 Linux 内核

首先请正确选择板卡配置文件

| 版本 | 配置文件 |
|--------|-------------------------|
| SDK4.3 | JN-industrPi_defconfig |
| SDK5.3 | JN-IndustriPi_defconfig |

然后执行如下命令去编译 Linux 内核

```
cd ~/ti-processor-sdk-linux-am57xx-evm-xx
```

```
cd board-support_x.x.x.x/linux-4.14.79+gitAUTOINC+c669d52447-ge669d52447
```



```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- JN-IndustriPi_defconfig
```

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- zImage -j4
```

编译时间约 10 分钟，编译成功后，在 arch/arm/boot/目录下生成名为 zImage 的 Linux 内核镜像文件。

注意：参数-j4 表示使用 4 线程进行编译。

3. 编译设备树文件

从 3.8 内核开始，每个板卡都有一个内核所需的设备树二进制文件。

下面为 IndustriPi 的设备树文件

| 版本 | 配置文件 |
|--------|---------------------------|
| SDK4.3 | JN-industrPi.dts |
| | JN-industrPi-common.dtsi |
| SDK5.3 | JN-IndustriPi.dts |
| | JN-IndustriPi-common.dtsi |

要构建设备树二进制文件，找到正在使用的板卡 dts 文件，并将.dts 扩展名替换为.dtb，

然后执行如下命令去编译生成设备树文件

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- JN-IndustriPi.dtb
```

编译时间约 1 分钟，编译成功后，在 arch/arm/boot/dts/目录下生成名为 JN-IndustriPi.dtb 的设备树二进制文件。

注意：也可以直接执行当前目录下 build.sh，去编译生成 zImage 和设备树二进制文件。 SDK4.3 和 SDK5.3 二进制文件名字不同，请参考 build.sh。

4. 编译模块

默认情况下，SDK 中使用的 Linux 驱动并为集成到内核镜像(zImage)中。

执行如下命令去配置新配置文件。

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig
```

或执行当前目录下的 buildmenu.sh

配置完成后，编译模块。

```
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- modules
```

或执行当前目录下的.buildmodules.sh

编译成功后，将在对应驱动目录下生成后缀为.ko 文件。

注意：

执行 buildmenu.sh 在图形界面选 M 为模块、选*为编译到内核中，保存退出后。

替换工业派配置文件: cp .config arch/arm/configs/JN-IndustriPi_defconfig

| 版本 | 配置文件 |
|--------|-------------------------|
| SDK4.3 | JN-industrPi_defconfig |
| SDK4.3 | JN-IndustriPi_defconfig |

如果执行 menuconfig 时出错

```
ce43c71ae$ ./buildmenu.sh
HOSTCC scripts/kconfig/mconf.o
In file included from scripts/kconfig/mconf.c:23:0:
scripts/kconfig/xdialog/dialog.h:38:20: fatal error: curses.h: 没有那个文件或目录
compilation terminated.
scripts/Makefile.host:124: recipe for target 'scripts/kconfig/mconf.o' failed
make[1]: *** [scripts/kconfig/mconf.o] Error 1
Makefile:546: recipe for target 'menuconfig' failed
make: *** [menuconfig] Error 2
```

执行如下命令去安装 curses 库。

sudo apt-get install libncurses-dev

2.7 固件更新

2.7.1 更新 SD 卡

假设您已经参考 2.8 章节制作好一张 SD 卡，您需要将 MLO 和 u-boot.img 文件复制到 boot 分区，将 zImage 和 JN-IndustriPi.dtb 文件复制到 rootfs 分区的 boot 目录下。

- 1. 使用读卡器将 MicroSD Card 连到 Ubuntu PC 上
- 2. 查看 MicroSD Card 卡挂载信息

Host:~# df

```
wangnan@wangnan:~$ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev            991788         4    991784   1% /dev
tmpfs           201068       1328    199740   1% /run
/dev/sda1       305506552 190993368  98971324  66% /
none              4           0         4    0% /sys/fs/cgroup
none             5120         0      5120    0% /run/lock
none          1005320       152   1005168   1% /run/shm
none           102400        60    102340   1% /run/user
/dev/sdb1       307534288 228757024  63132372  79% /back
/dev/sdc1        77619        716    76903   1% /media/wangnan/boot
/dev/sdc2       7414160     6657556   356940  95% /media/wangnan/rootfs
```

如上图可知：

MicroSD Card 卡 boot 分区挂载目录为/media/xxx/boot

MicroSD Card 卡文件系统分区挂载目录为/media/xxx/rootfs

- 3. 拷贝镜像文件

```
cp MLO u-boot.img /media/xxx/boot  
cp zImage JN-IndustriPi.dtb /media/xxx/rootfs/boot
```

2.7.2 更新 emmc

IndustriPi 板卡自带 Linux 系统。启动板卡，您需要将 MLO 和 u-boot.img 文件复制到 boot 分区，将 zImage 和 JN-IndustriPi.dtb 文件复制到 rootfs 分区的 boot 目录下。

1. 请参考 1.1 章节启动板卡。
2. 查看 emmc 分区信息

```
root@am57xx-evm:~# df
```

```
root@am57xx-evm:~# df
```

| Filesystem | 1K-blocks | Used | Available | Use% | Mounted on |
|----------------|-----------|---------|-----------|------|----------------------|
| /dev/root | 7289224 | 4924632 | 1971276 | 71% | / |
| devtmpfs | 322300 | 4 | 322296 | 0% | /dev |
| tmpfs | 413436 | 8 | 413428 | 0% | /dev/shm |
| tmpfs | 413436 | 9752 | 403684 | 2% | /run |
| tmpfs | 413436 | 0 | 413436 | 0% | /sys/fs/cgroup |
| tmpfs | 413436 | 4 | 413432 | 0% | /tmp |
| tmpfs | 51200 | 112 | 51088 | 0% | /var/volatile |
| tmpfs | 16384 | 0 | 16384 | 0% | /media/ram |
| /dev/mmcblk1p1 | 77619 | 767 | 76853 | 1% | /run/media/mmcblk1p1 |
| tmpfs | 82688 | 0 | 82688 | 0% | /run/user/0 |

如图所知，emmc 的 boot 分区挂载目录为/run/media/mmcblk1p1

3. 拷贝镜像文件

```
cp MLO u-boot.img /run/media/mmcblk1p1  
cp zImage JN-IndustriPi.dtb /boot
```

2.8 制作 SD 启动卡

1. 准备

开发环境：ubuntu-14.04 或 ubuntu-16.04 版本开发环境。

硬件：一张 MicroSD 卡（容量至少 8G，Class 10）以及读卡器。

2. 烧写包简介

固件：JN-industriPi_programming_x.x.x.x.tar.gz。

解压：tar xvf JN-industriPi_programming_x.x.x.x.tar.gz

烧写包有以下内容：

烧写脚本：mkmmc-am57xx.sh

uboot 镜像：MLO u-boot.img

内核镜像: zImage

设备树二进制文件: JN-IndustriPi.dtb

文件系统: tisd-k-rootfs-image-am57xx-evm.tar.xz

文件系统补丁: fs_patch

3. 镜像烧写

运行如下命令:

```
./mkmmc-am57xx.sh /dev/sdb MLO u-boot.img zImage JN-industriPi.dtb
```

tisd-k-rootfs-image-am57xx-evm.tar.xz

当打印 mkmmc-am57xx success 时表示烧写成功。

注意:

ls /dev/sd* 查看设备节点:

```
/dev/sda /dev/sda1 /dev/sda2 /dev/sda5 /dev/sdb /dev/sdb1 /dev/sdb2
```

设备节点/dev/sda 对应 ubuntu 根文件系统, 可以确定设备节点/dev/sdb 对应 MicroSD 卡。/dev/sdb1 和/dev/sdb2 为分区设备号, 所以参数为/dev/sdb。

2.9 烧写 emmc

1 拷贝固件 JN-IndustriPi_programming_x.x 到 Micro SD 启动卡文件系统分区的主目录

2 将 Micro SD 启动卡连接到工业派 P62 接口, 参照第 1 章启动板卡。

3 烧写 emmc

运行如下命令:

```
./mkmmc-am57xx.sh /dev/mmcblk1 MLO u-boot.img zImage JN-industriPi.dtb
```

tisd-k-rootfs-image-am57xx-evm.tar.xz

烧写时间大概 8 分钟, 打印信息如下图 2-8 所示, 表示 eMMC 烧写成功。

```
All data will be cleared [y/n]
y
Upload the partition
Create the partition
Format the partition
Fill up the partition
mkmmc-am57xx success
```

第 3 章 图形显示框架

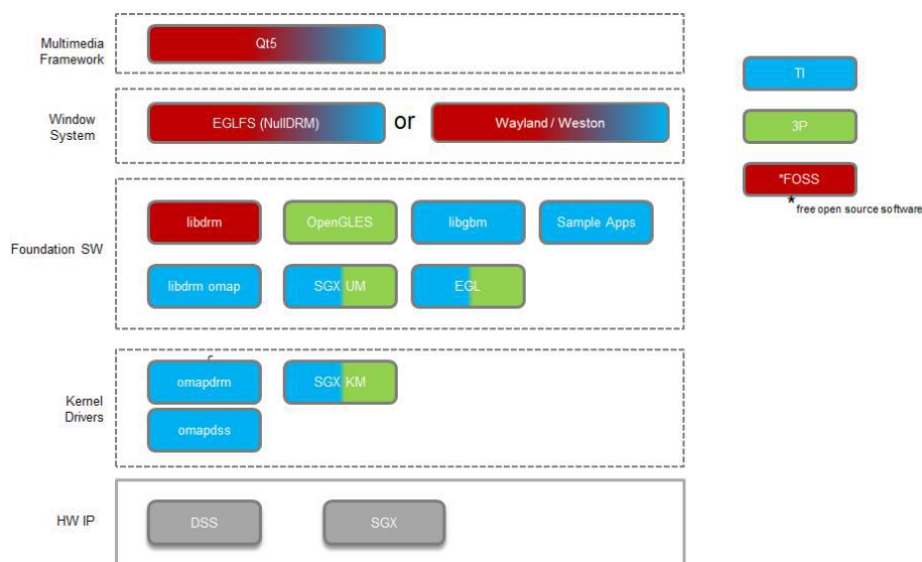
AM57xx 采用专用硬件 SGX 来加速 3D 操作

下表为 SDK 支持的 SGX 核心信息：

| SGX 核心 | SGX 版本 | SGX 频率(MHz) |
|--------|--------|-------------|
| SGX544 | 1.1.6 | 536 |

当前版本的 SGX DDK 提供了 OpenGL ES2.0 和 EGL 库，这些库用于 SDK 中的图像堆栈，如 QT5 和 Wayland/Weston,目前不支持基于 Mesa-EGL 的应用程序。

软件框架图：



通过 TI Processor Linux 自带 Matrix 用户界面可以演示以下 3D 图形。

| 演示案例 | 描述 |
|--------------|-----------------------|
| ExampleUI | 该演示展示了如何有效渲染界面元素 |
| ChameleonMan | 此演示结合凹凸贴图显示矩阵角色 |
| KMS Cube | 该演示展示了如何渲染和显示多色旋转立方体 |
| Coverflow | 这是一个 Coverflow 样式效果演示 |
| Navigation | 该演示展示了一个导航软件的算法演示 |

3.1 QT5 图形框架

Qt 是一个跨平台的 C++ 应用程序开发框架，主要用于开发图形应用程序。

从 Qt5.0 开始，Qt 不再支持 QWS 窗口系统，取而代之的新机制是 Qt 平台抽象-QPA(Qt Platform Abstraction)，QPA 使得 Qt 对不同平台的支持变得更加灵活，当需要支持一个新平台时，只需为该平台编写一个 QPA 插件，运行时通过“-platform”来制定 QPA 插件（qtbase/src/plugins/platforms），如果不指定就默认使用 QPA 插件，比如

```
./qt -platform linuxfb
```

```
./qt -platform eglfs
```

具体开发教程请参考《QT 开发教程 V1.0》手册。

注意：如果使用 LinuxFB、EGLFS、KMS 平台，需要关闭 weston 服务：/etc/init.d/weston stop

如果直接在 weston 显示系统上显示 qt，则直接执行 qt 程序 ./qt。

第 4 章 多媒体框架

GStreamer 是用来构建流媒体应用的开源多媒体框架,目标是简化音/视频应用程序的开发。采用了基于插件 (plugin) 和管道 (pipeline) 的体系结构,框架中的所有的功能模块都被实现成可以组装的组件 (component),并且在需要的时候能够很方便地安装到任意一个管道上,由于所有插件都通过管道机制进行统一的数据交换,因此很容易利用已有的各种插件“组装”出一个功能完善的多媒体应用程序。

TI GStreamer 插件支持列表

- Ducati Decoding and Encoding

- ducatih264dec

- ducatimpeg4dec

- ducatimpeg2dec

- ducative1dec

- ducatijpegdec

- ducatih264enc

- ducatimpeg4enc

- Ducati VPE

- vpe

- ducatih264decvpe

- ducatimpeg2decvpe

- ducatimpeg4decvpe

- ducatijpegdecvpe

- ducative1decvpe

- DSP Image Processing

- dsp66videokernel

- ARM HEVC Decoding

- h265dec

Gstreamer 编解码的使用说明:

1. h264 编码

将测试视频编码保存为 h264 文件

```
gst-launch-1.0 -v videotestsrc pattern=18 ! 'video/x-raw, format=(string)YUY2, width=(int)1280, height=(int)720, framerate=(fraction)60/1' !vpe num-input-buffers=8 ! 'video/x-raw, format=(string)NV12, width=(int)1280, height=(int)720,framerate=(fraction)60/1' ! ducatih264enc bitrate=1000 ! filesink location=test.h264
```

2. h264 解码

解码显示 h264 文件

```
gst-launch-1.0 -v filesrc location=test.h264 ! queue ! h264parse ! ducatih264dec ! waylandsink
```

3. AAC 音频解码

解码音频文件格式为 aac

```
gst-launch-1.0 filesrc location=test.aac ! faad ! alsasink
```

4. 采集编码为 MP4 文件

```
gst-launch-1.0 -e videotestsrc pattern=18 ! 'video/x-raw, format=(string)NV12, width=(int)640, height=(int)480, framerate=(fraction)30/1' ! vpe num-input-buffers=8 ! 'video/x-raw, format=(string)NV12, width=(int)1280, height=(int)720' ! queue ! ducatimpeg4enc bitrate=4000 ! queue ! mpeg4videoparse ! qtmux ! filesink location=test.mp4
```

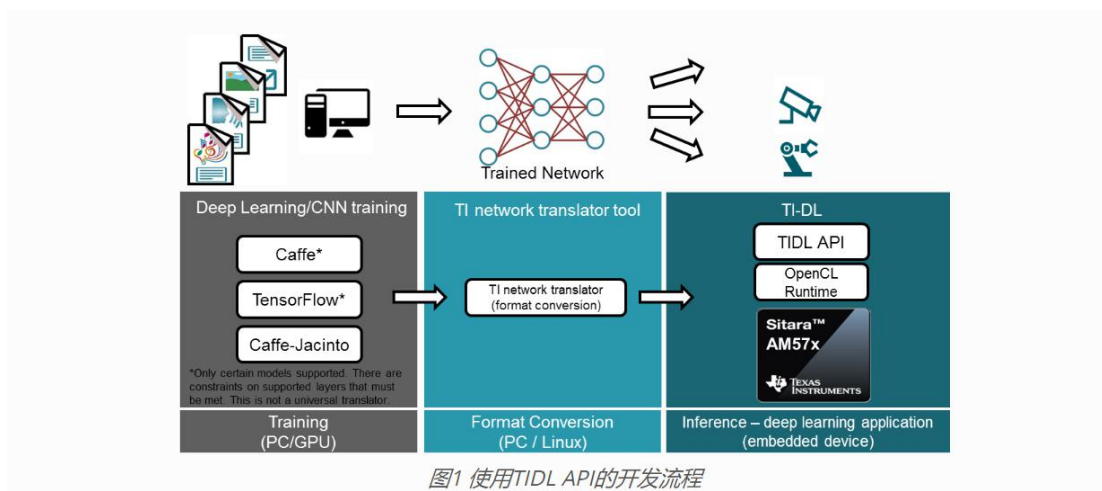
5. 解码 mp4 文件

```
gst-launch-1.0 -v filesrc location=test.mp4 ! qtdemux name=demux demux.video_0 ! queue ! mpeg4videoparse ! ducatimpeg4dec ! waylandsink
```

第5章 TIDL 框架

TI 深度学习（TIDL）利用 TI 专有 API 使应用程序运行在 EVE 和 C66x DSP 计算引擎上，拥有高度优化的 CNN / DNN 从而实现深度学习。TIDL API 显着改善了用户的开箱即用深度学习体验，使他们能够专注于整体用例。不必花时间研究 Arm®↔DSP/ EVE 通信的机制。API 允许客户轻松集成 OpenCV 等框架并快速构建深度学习应用程序原型。拥有高度优化的 CNN / DNN 从而实现深度学习。

开发流程：



深度学习包括两个阶段：开发阶段的训练和部署阶段的推理。训练涉及设计神经网络模型，通过神经网络运行训练数据以调整模型参数。推理采用包含参数的预训练模型。使用 Caffe / TensorFlow 等框架完成。一旦被训练，TIDL 转换器工具可用于将参数转换。

目前，TIDL 软件主要使用离线预训练模型进行卷积神经网络推理，存储在设备文件系统中（无需在目标设备上进行）。使用 Caffe 或 Tensorflow-slim 框架训练的模型导入和转换

TI 官网资料：

<http://software-dl.ti.com/processor-sdk-linux/esd/docs/latest/linux/Foundational Components TIDL.html>

<https://training.ti.com/texas-instruments-deep-learning-tidl-overview>

导入工具：文件系统 /usr/bin/tidl_model_import.out

或 linux-devkit/sysroots/x86_64-arago-linux/usr/bin

TIDL API 示例：文件系统/usr/share/ti/tidl/

源码目录：

| example | Link |
|-------------------------------------|---|
| Imagenet Classification | 图像分类 |
| Segmentation | 像素分割 |
| SSD_multibox | 单发多盒检测,要求 AM57x 处理器同时具有 EVE 和 C66X, 工业派不支持。 |
| Test | 单元测试 |
| Classification with class filtering | tidl 矩阵图形用户界面的演示 |

案例编译:

```
cd ti-processor-sdk-linux-am57xx-evm-05.03.00.05-Linux
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- tidl-examples
```

5.1 TIDL 案例

1、tidl_classification 命令讲解

可选参数:

- c 配置文件路径
- d <DSP 内核数> (0-2)
- e <EVE 核心数> (0-2)
- g <1| 2> 图层数
- l 模型中所有类别
- s 包含所选类的字符串列表
- i 视频输入 (用于摄像头设备 0,1 或者视频流)

2、图像识别检测。

```
cd /usr/share/ti/tidl/examples/classification
./tidl_classification -g 1 -d 1 -e 0 -l ./imagenet.txt -s ./classlist.txt -i 1 -c ./stream_config_j11_v2.txt
```

imagenet.txt 是配置文件 stream_config_j11_v2.txt 中指定模型可以检测到的所有类列表。检测模型的列表在 ./classlist.txt 中指定。

注意: AM5708 是单核 DSP, 不带 EVE 核。 所以-d 为 1、 -e 为 0。