1.产品介绍

EdgeBoard 是基于FPGA打造的嵌入式AI解决方案及基于此方案实现的系列硬件，与AI Studio/EasyDL等模型定制平台深度打通，具有高性能、高通用、低成本、易开发等四大优点，适用于开发验证、产品集成、科研教学、项目落地等应用方向，以及安防监控、工业质检、医疗诊断、农作物生长监控、无人驾驶、无人零售等应用场景。

本文以FZ9A计算卡为例（盒子和其它版本计算卡同样适用），介绍如何使用EdgeBoard进行深度学习应用开发，官方团队联系方式见文档末尾。

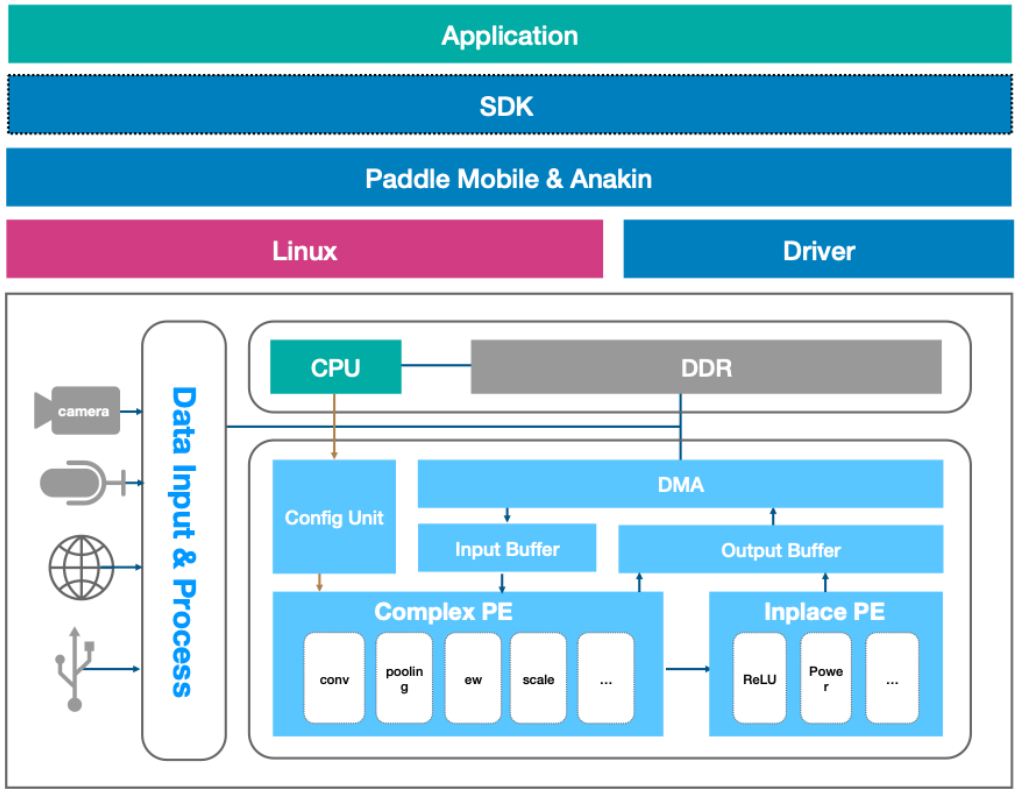
1.1 硬件介绍

高性能计算板卡EdgeBoard是基于Xilinx Zynq UltraScale+ MPSoCs 开发平台的开发板，EdgeBoard采用Xilinx FPGA核心处理器将多核ARM Cortex-A53和FPGA可编程逻辑集成在一颗芯片上。在EdgeBoard高性能计算板卡上搭载了丰富的外部接口和设备，方便用户的使用和功能验证。

EdgeBoard高性能计算板卡分为上下主板相结合的设计理念，分为Main-Board和Power-Board，在整板对外接口最大化基础上大大缩小主板的尺寸，方便客户直接将EdgeBoard集成到整机成品中。

EdgeBoard内含模型嵌入工具包、AI加速工具包、嵌入式计算卡成品，其软硬一体的完整架构造具有强悍的终端加速性能。赛灵思FPGA高性能的加速引擎提供3.6Tops的强大AI算力，性能50倍于终端CPU、Resnet50实测可达到60FPS处理速度，且方案商可以根据需要输入百度大脑预置或自定制模型。

1.2 软件介绍



EdgeBoard基于linux系统，整个开发过程就是一个linux应用程序的开发。应用程序获取视频输入，调用预测库加载模型，调度模型，驱动加速模块进行计算，加速模型运行，获得运行结果。

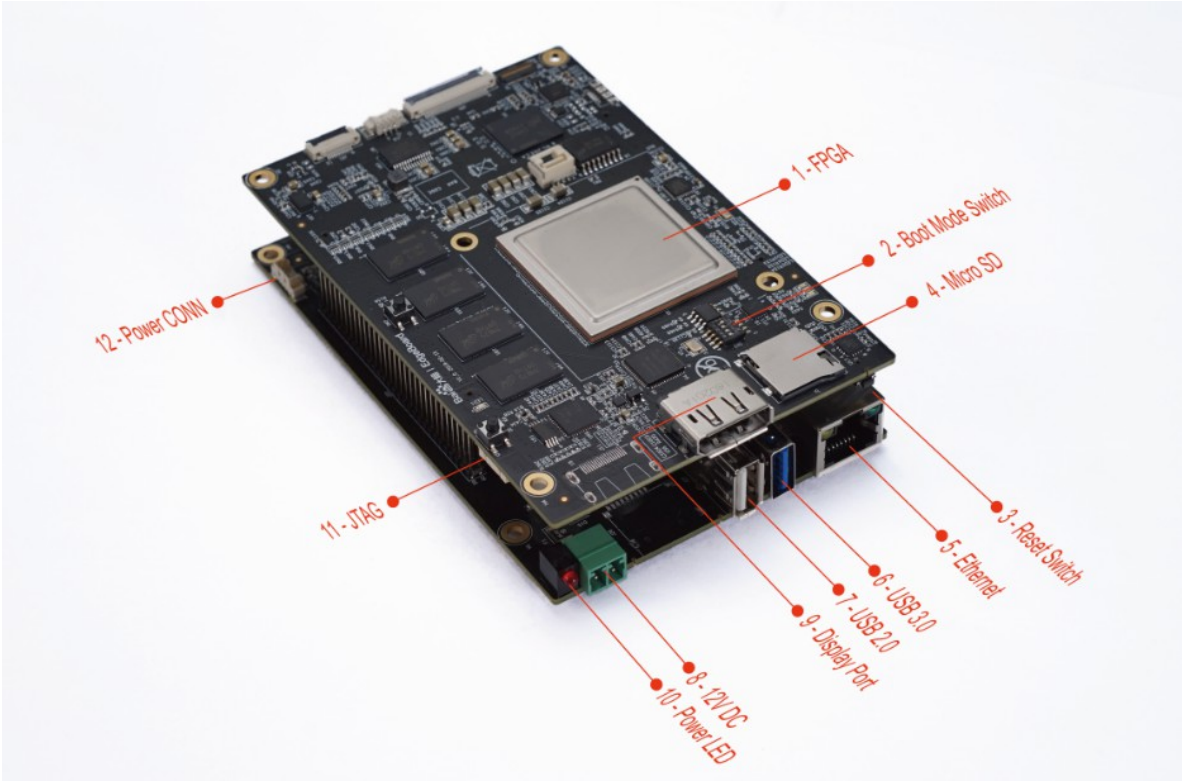
1.3 常用模型在EdgeBoard FZ9A上的性能数据

网络	输入尺寸	单帧耗时
resnet50	224 x 224	16ms
resnet101	224 x 224	33ms
mobilenet-v1	224 x 224	7ms
vgg-ssd	300 x 300	59ms
inception-v2	299 x 299	19ms
inception-v3	299 x 299	31ms
inception-v4	299 x 299	61ms
mobilenet-ssd	224 x 224	19ms
mobilenet-ssd-640	640 x 640	59ms
yolo-v3	416 x 416	80ms

注：EdgeBoard软核仍在持续升级，性能将同步提升。相同网络结构不同版本对算力要求不同，如有具体项目应用，可联系官方团队申请定制优化

2.启动与连接

2.1 启动准备



EdgeBoard FZ9A计算卡示意图1

EdgeBoard FZ9A计算卡示意图2

1. 保证配套的系统TF卡已经插到开发板接口，上图4-MicroSD Card；
2. 使用配套电源给EdgeBoard供电，上图8-12V DC
3. EdgeBoard支持两种调试方式，网络调试和串口调试，推荐使用网络SSH连接方式调试更加方便快捷（详见后文）
   1）使用SSH网络服务登录系统,详见后文
   2）使用串口登录系统（可选）

4. EdgeBoard启动完成后输入login&password为root&root，就可以进到EdgeBoard的系统，运行系统自带的Sample了，调试示例

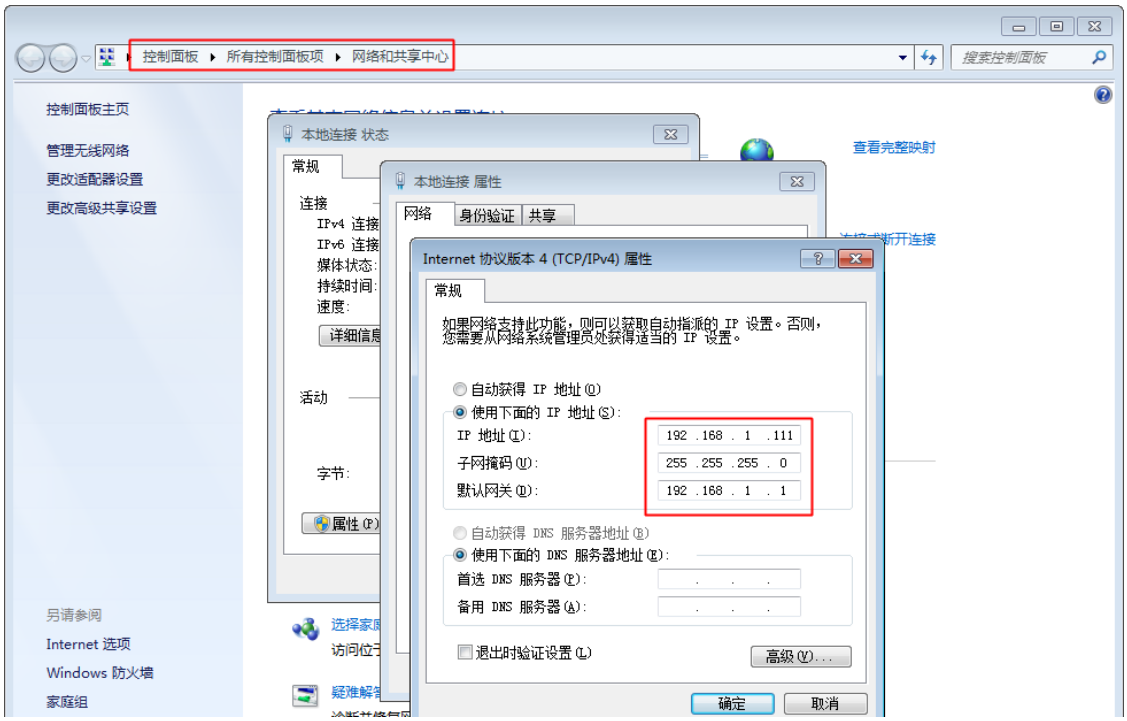
2.2 连接方式一：SSH连接

- EdgeBoard出厂默认参数为静态ip=192.168.1.254，netmask=255:255:255:0，gateway=192.168.1.1
- 硬件连接方法：使用网线一端连接EdgeBoard，另一端连接host电脑或者路由器，设置电脑或路由器ip和EdgeBoard在同一网段，即可使用SSH登录

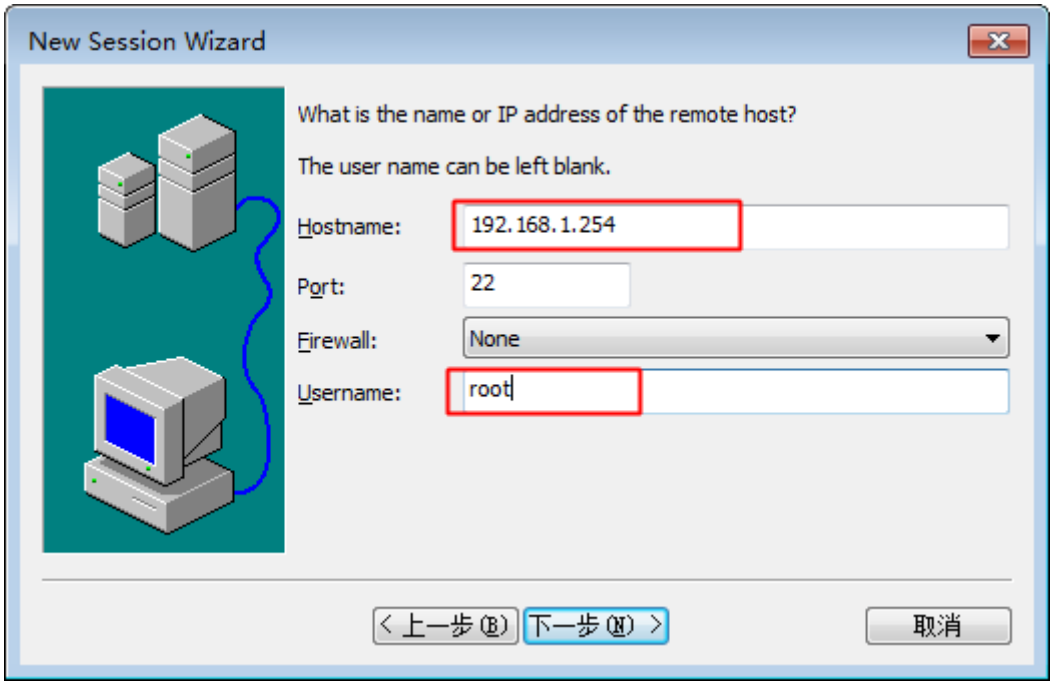
具体步骤如下：

2.2.1 Windows使用SecureCRT网络服务连接网口方法

1. 安装调试工具，推荐SecureCRT工具（可百度搜索安装）；
2. 配置电脑或路由器ip和设备在同一网段下,当电脑和设备直连时需要手动设置电脑ip，打开网络和共享中心-->本地连接-->属性-->Internet 协议版本4，手动配置ip地址：192.168.1.111，子网掩码：255.255.255.0，默认网关：192.168.1.1,如下图所示。



3. 在secure CRT中新建窗口，connect-->New Session-->Protocol选择SSH2,点击下一步，Hostname为EdgeBoard的ip，出厂默认192.168.1.254，port=22，下一步，完成，在弹窗中输入username=root，password=root，即可进入系统。



2.2.2 MAC使用SSH网络服务连接网口方法

1. 配置电脑（或者路由器）ip为192.168.1.xxx（1<xxx<253），保证电脑和EdgeBoard的ip在同一网段
   配置步骤：系统偏好设置-->网络-->高级-->TCP/IP。
   IPv4配置示例：手动，IPv4地址：192.168.1.111，子网掩码：255.255.255.0，路由器：192.168.1.1

2. **打开Terminal:** Launchpad->其它(文件夹)->终端（即Terminal程序）
3. 在Terminal中输入ssh root@ip (EdgeBoard的IP地址)，默认地址为 ssh root@192.168.1.254，然后再输入login&password为root&root,即可登录EdgeBoard系统

2.3连接方式二：串口连接（备选）

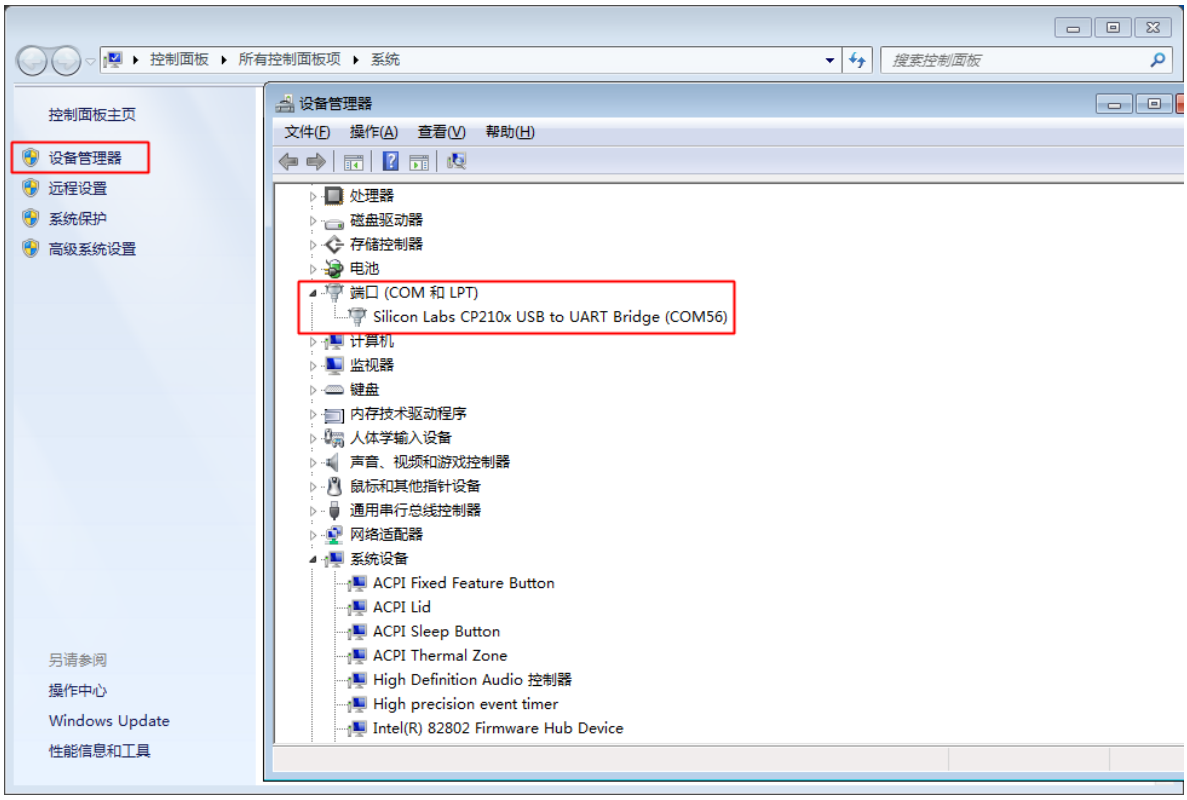
如果出现ssh连接不上，或者（设备ip动态获取后）需要查看ip，需要使用串口进入设备的控制台

可使用micro usb数据线连接EdgeBoard的USB UART接口（详见示意图17-USB UART），使用电脑连接EdgeBoard系统。

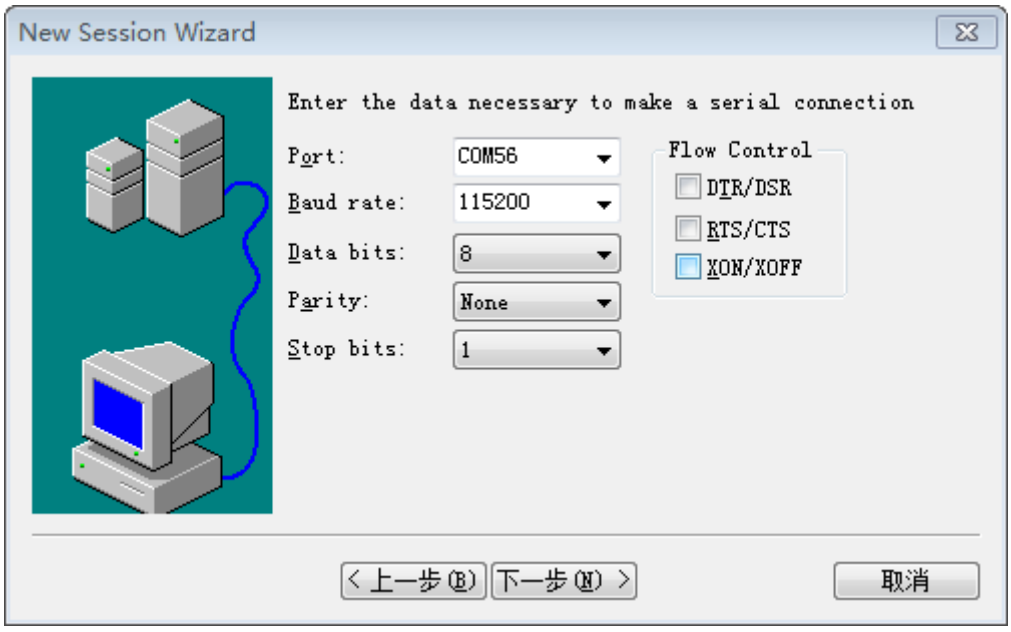
2.3.1Windows使用SecureCRT连接串口方法

1.安装SecureCRT软件和串口驱动 CP210x_Windows_Drivers（初次使用需安装驱动，安装包可百度搜索）

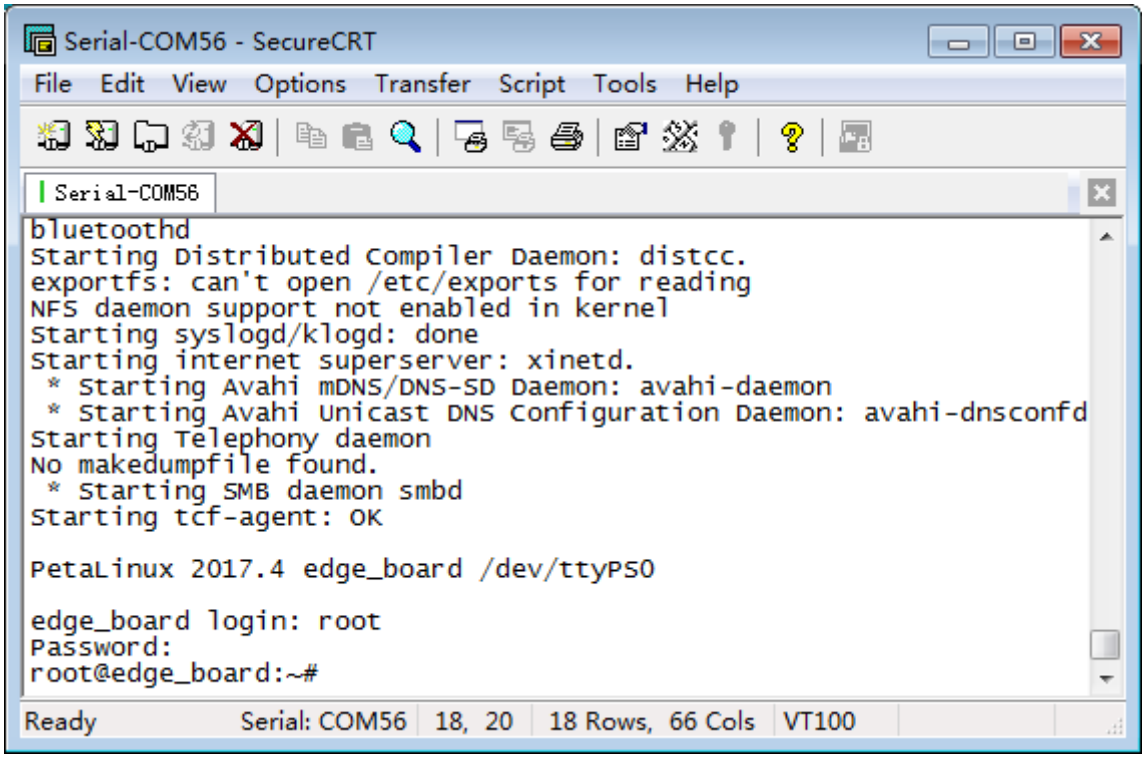
2.保证电脑已连接EdgeBoard的USB UART接口（详见示意图接口17-USB UART），【右击“我的电脑”->属性->设备管理器】查看设备管理器中映射的端口号，如图示，端口号为COM56



3.打开SecureCRT，新建窗口connect->New Session->Protocol选择Serial，波特率选择115200，Flow Control不选，如下图所示



4.点击【完成】-【Connect】按钮，SecureCRT会连接到计算盒上的串口，上电后即可看到启动信息，待启动完成后输入用户名和密码root/root,即可进入设备系统。如下图所示



2.3.2 MAC使用minicom连接串口方法

1.在电脑中安装串口驱动：SiLabsUSBDriverDisk.dmg（初次使用需安装，可百度搜索安装方法）
2.mac安装minicom工具（可百度搜索安装方法）
3.**打开Terminal:** Launchpad->其它(文件夹)->终端（即Terminal程序），在terminal终端输入minicom -s 进行配置
4.配置内容如下，配置完成后，连接上EdgeBoard，在terminal终端输入minicom即可。

- 选择Serial port setup，配置如下：
  A - Serial Device : /dev/cu.SLAB_USBtoUART
  B - Lockfile Location : /usr/local/Cellar/minicom/2.7/var
  C - Callin Program :
  D - Callout Program :
  E - Bps/Par/Bits : 115200 8N1
  F - Hardware Flow Control : No
  G - Software Flow Control : No
- Save setup as dfl

3.调试设备

3.1 更改设备网络配置方法

设备出厂默认为静态ip地址192.168.1.254，如果多个设备同时连到同一个局域网，则需要更改设备为不同的ip地址，或者改为动态获取ip的方式，网络配置文件路径为/etc/network/interfaces

```
//打开并编辑interfaces文件
vim /etc/network/interfaces
```

静态ip配置

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.1.254
netmask 255.255.255.0
gateway 192.168.1.1
broadcast 192.168.1.255
```

动态ip配置

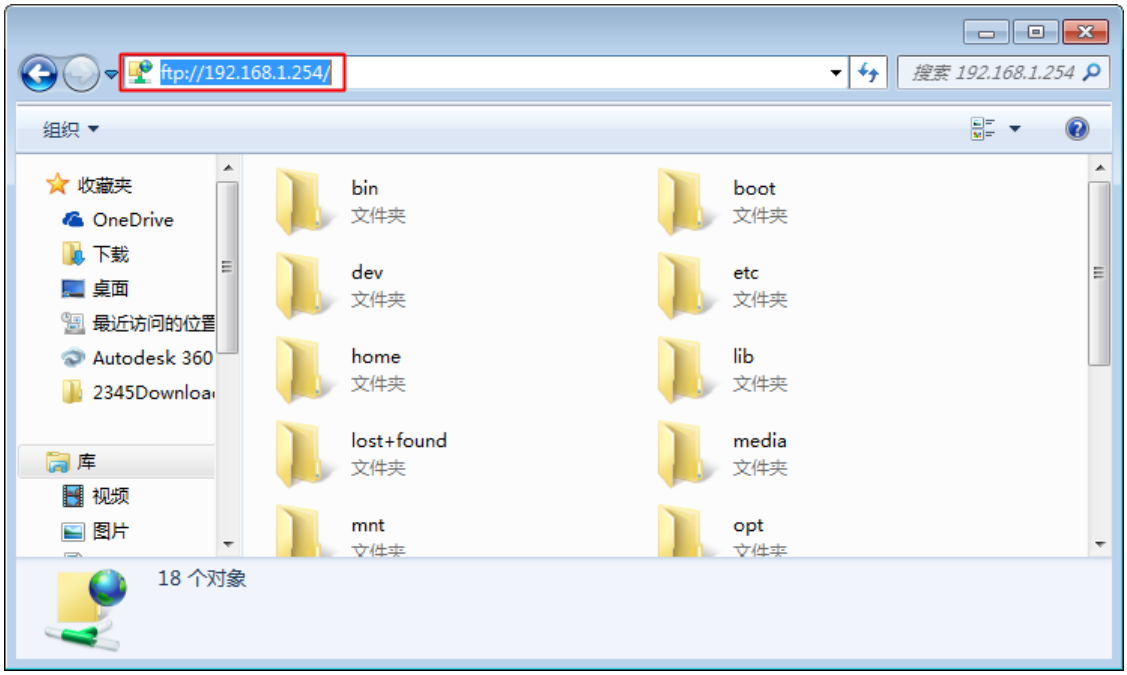
```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp
```

3.2 文件拷贝

EdgeBoard支持ssh、samba、ftp等网络协议，可轻松通过网络进行数据通讯以及文件拷贝的工作，使用这一功能在后面软件升级、用户定制时都会有广泛的应用。

3.2.1 通过FTP实现文件拷贝（适用于windows系统）

1. 【Windows+R快捷键组合，输入ipconfig】ifconfg命令查看设备ip，保证设备ip和windows电脑的ip在同一网段，在文件夹输入框里直接输入ftp://192.168.1.254 。根据提示输入用户名root，密码root。即可进入设备系统



2. 打开home-->root-->workspace目录，workspace是root用户下应用程序所在的目录，直接拷贝文件到workspace，或者从workspace中拷贝文件到电脑即可。

3.2.2 通过samba协议实现文件拷贝（适用于Mac OS系统）

1. 配置并保证设备ip和MAC的ip在同一网段（参见上文：MAC使用SSH网络服务连接网口方法）
2. 配置完成后，点击Finder-->前往-->连接服务器， 输入smb://ip，例如smb://192.168.1.254,用户名root，密码root。

3. finder中出现设备的文件目录，打开home-->root-->workspace目录，workspace是root用户下应用程序所在的目录，可以直接通复制粘贴命令进行电脑和设备间文件的拷贝。

3.3 系统目录介绍

内容	目录	备注
paddle-mobile	/home/root/workspace/paddle-mobile	paddle-mobile预测库
driver	/home/root/workspace/driver	驱动文件
sample	/home/root/workspace/sample	示例代码
tools	/home/root/workspace/tools	调试工具

3.4 运行Sample

当您连接上EdgeBoard后，可以运行我们提供的深度学习示例，位于home/root/workspace/sample。

示例	说明
classification	分类模型示例
detection	目标检测模型示例

3.4.1 分类模型示例

读取一张本地图片，调用模型进行推理，并输出结果。

考虑到简单通用性，该示例，从json文件中读取模型和图片信息，加载并执行。 执行时需要指定 相应的配置文件。如

```
./image_classify ../configs/resnet50/drink.json
```

工程目录结构:

```
├── CMakeLists.txt // cmake 工程配置文件。
├── include
|   └── io // paddle_mobile 头文件目录
|       ├── paddle_inference_api.h
|       ├── type_define.h
|       └── types.h
├── configs // 配置文件目录
|   ├── Inceptionv2
|   |   └ zebra.json 斑马图片的配置
|   ├── Inceptionv3
|   └── resnet50
├── lib
|   └── libpaddle-mobile.so
├── models // 模型文件目录
|   ├── Inceptionv2
|   ├── Inceptionv3
|   └── resnet50
|── src
|   ├── json.hpp // json 解析库
```

```
|   └── image_classify.cpp // 图片推理示例
└── README.md
```

下面是配置文件示例。

```
{
    "model":"../models/resnet50",
    "combined_model":true, //
    "input_width":224,
    "input_height":224,
    "image":"../models/resnet50/drink.jpeg",
    "mean":[104,117,124],
    "scale":1,
    "format":"BGR"
}
```

key	value
model	模型目录存放的位置
combined_model	是否为融合的模型，只有两个文件的是融合模型
input_width	输入网络的图片尺寸 输入图片会缩放到该大小
input_height	输入网络的图片尺寸
image	进行分类的图片输入
mean	平均值
scale	输入网络前预算处理为 (x - mean) * scale
format	网络所需要的格式，OpenCV默认是BGR

其它的分类网络也可以通过添加/修改 配置文件实现，无须修改代码。

1.加载驱动,系统启动后加载一次即可（也可以加系统启动脚本）

```
insmod /home/root/workspace/driver/fpgadrv.ko
```

2.编译示例，EdgeBoard上具有编译能力，进入到sample/classification示例的build目录下进行编译

```
// 打开示例目录
cd /home/root/workspace/sample/classification
// 如果没有build目录，创建一个
mkdir build
cd build
rm -rf *
// 调用cmake 创建 Makefile
cmake ..
// 编译工程。
make
```

编译结束后会在build 目录生成如下几个文件。
image_classify 读取本地图片推理示例。
video_classify 读取摄像头数据进行推理，要连接摄像头(USB Camera，MIPI CSI Camera等)才能使用。如需显示结果还要连接DP显示器或者HDMI显示器或者VGA显示器。

对于一些需要MIPI摄像头支持的嵌入式应用场景的设备，也提供了一块MIPI镜头模组作为配件，并已经做了系统和软硬件上的支持，可以作为开发中的评估使用。(EdgeBoard已经适配的MIPI Camera AR20-7022-F0可以在百度AI市场上进行购买：https://aim.baidu.com/product/356080a4-d7f3-4ef9-9e85-abea8a5c49bc)

3.执行示例

```
使用图片推理
./image_classify ../configs/resnet50/drink.json

使用视频推理，必须保证USB摄像头和显示器同时连接才可正确运行
startx  //打开桌面环境
./video_classify ../configs/resnet50/drink.json
```

3.4.2 目标检测示例

和分类不同，物体检测除了能知道物体的类型，还能检测出物体所在的位置坐标。物体检测也分了两个示例，一个是在图片上检测物体，并绘制出坐标信息。还有通过摄像头采集视频，检测在屏幕上绘制坐标信息。

工程目录结构:

```
├── CMakeLists.txt // cmake 工程配置文件。
├── include
|   └── io // paddle_mobile 头文件目录
|       ├── paddle_inference_api.h
|       ├── type_define.h
|       └── types.h
├── configs // 配置文件目录
|   ├── yolov3
|   └── ssd
├── lib
|   └── libpaddle-mobile.so
├── models // 模型文件目录
|   ├── yolov3
|   └── ssd
|── src
|   ├── json.hpp // json 解析库
|   └── image_detect.cpp // 图片推理示例
└── README.md
```

下面是配置文件示例。

```
{
    "model":"../models/ssd",
    "combined_model":true, //
    "input_width":224,
    "input_height":224,
    "image":"../models/ssd/screw.jpg",
    "mean":[104,117,124],
    "scale":1,
    "format":"BGR"
}
```

key	value
model	模型目录存放的位置
combined_model	是否为融合的模型，只有两个文件的是融合模型
input_width	输入网络的图片尺寸 输入图片会缩放到该大小
input_height	输入网络的图片尺寸
image	进行分类的图片输入
mean	平均值
scale	输入网络前预算处理为 (x - mean) * scale
format	网络所需要的格式，OpenCV默认是BGR
threshold	信心值阈值

其它的分类网络也可以通过添加/修改 配置文件实现，无须修改代码。

1.加载驱动,系统启动后加载一次即可（也可以加系统启动脚本）

```
insmod /home/root/workspace/driver/fpgadrv.ko
```

2.编译示例，EdgeBoard上具有编译能力，进入到sample/detection示例的build目录下进行编译

```
cd /home/root/workspace/sample/detection
// 如果没有build目录，创建一个
mkdir build
cd build
rm -rf *
// 调用cmake 创建 Makefile
cmake ..
// 编译工程。
make
```

编译结束后会在build 目录生成如下几个文件。
image_classify 读取本地图片推理示例。
video_classify 读取摄像头数据进行推理，要连接摄像头(USB Camera，MIPI CSI Camera等)才能使用。如需显示结果需要连接DP显示器或者HDMI显示器或者VGA显示器,使用方法参见2.2.3

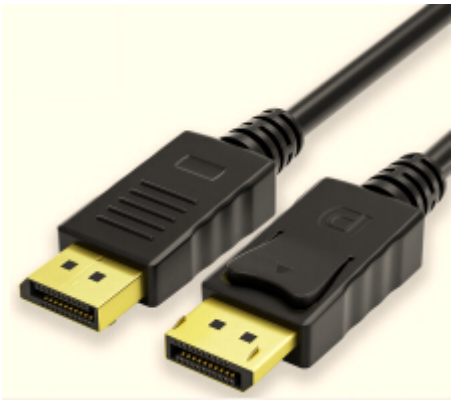
3.执行示例

```
使用图片推理
./image_detection ../configs/ssd/screw.json

使用视频推理，必须保证USB摄像头和显示器同时连接才可正确运行
startx  //打开桌面环境
./video_detection ../configs/ssd/screw.json
```

EdgeBoard FZ9A系列硬件平台提供的是DP(DisplayPort)输出接口，可以用来连接支持DP口的显示器或者连接HDMI接口的显示器或者VGA接口的显示器。

3.4.3 推理结果输出显示

1.DP接口显示器：使用公对公的DP转DP线连接EdgeBoard和DP显示器即可。



2.HDMI接口显示器：需要使用【注意：主动式DP公转HDMI母】转换线来连接EdgeBoard和HDMI显示器。参考链接1

3.VGA显示器：使用DP公转VGA母转接线连接EdgeBoard和VGA显示器即可。参考链接

提供的软件系统支持一个精简版的Linux桌面环境，用来实时显示程序运行的效果。
在上电使用前请确保显示器和Edgeboard已经通过DP连接线连接好，进入系统后默认是进入终端命令行的环境。可以通过以下命令进入和退出桌面环境

```
startx //打开桌面环境
stopx //关闭桌面环境
```

对于我们提供的演示示例，本身通过opencv已经支持将预测结果可视化的通过桌面窗口显示出来，运行方法参考2.2.1和2.2.2。

3.4.4 Video视频输入方式

Edgeboard支持USB、MIPI CSI、BT1120等协议的视频数据输入，可以作为各种场景的视频流的处理模块。比如在支持BT1120协议的IPC相机中，Edgeboard可以作为一个深度学习加速设备，IPC保持原有的正常功能。

EdgeBoard通过BT1120协议接收原始数据进行推理后，可以把结果通过串口、SPI传回IPC。类似的IPC模组后续将作为EdgeBoard配件上架AI市场。相应的IPC程序源码我们也会陆续开放。如果开发者自己有类似的设备，可以参考后面的硬件介绍，通过物理连接线连接。

对于一些需要MIPI摄像头支持的嵌入式应用场景的设备，也提供了一块MIPI镜头模组作为配件，并已经做了系统和软硬件上的支持（由于处理器资源限制，目前仅FZ9A系列产品支持MIPI应用，FZ5A产品如有MIPI需求请和我们联系），可以作为开发中的评估使用。(EdgeBoard已经适配的MIPI Camera AR20-7022-F0可以在百度AI市场上进行购买：https://aim.baidu.com/product/356080a4-d7f3-4ef9-9e85-abea8a5c49bc)

3.5 运行EasyDL平台模型预测示例

3.5.1 EasyDL的使用方式

EasyDL是一站式的深度学习模型训练和服务平台，提供可视化的操作界面，只需上传少量图片就可以获得高精度模型，具体可以参考EasyDL官网，通过EasyDL进行数据训练步骤如下：

3.5.1.1 选择训练类别

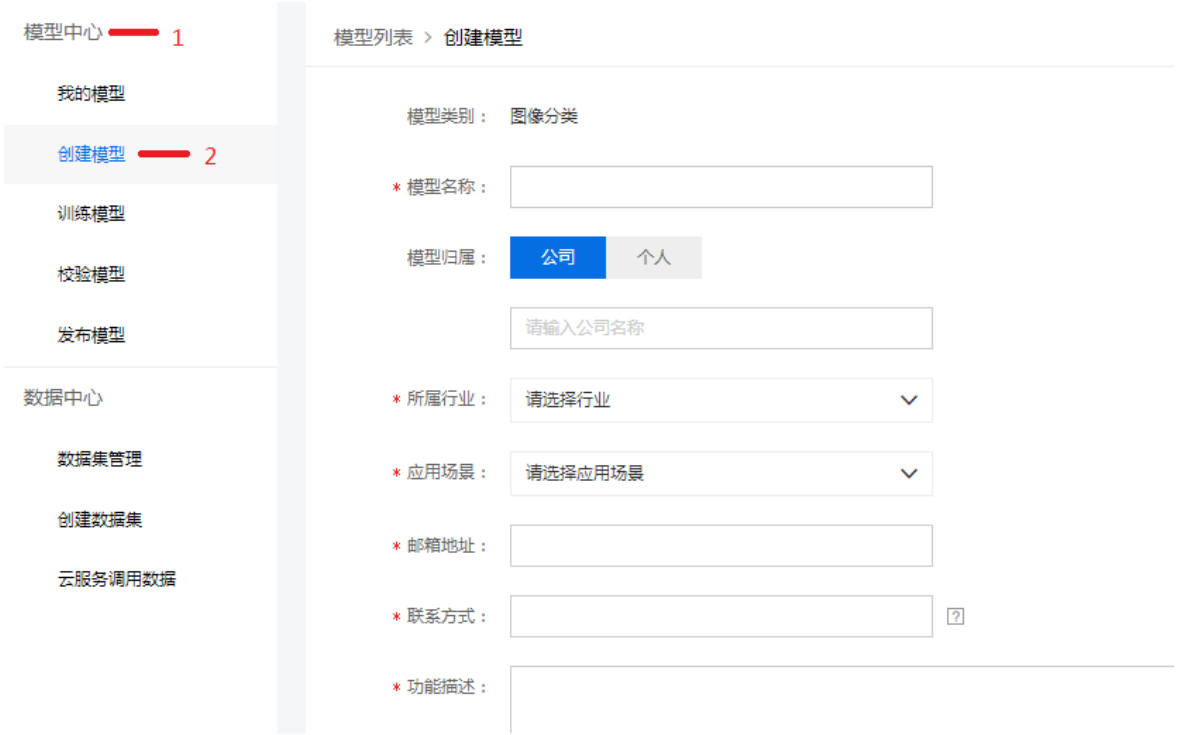
根据通用场景可以选择“图像分类”或者“物体检测”。

3.5.1.2 “图像分类”训练

选择“图像分类”，进入训练界面，选择“开始训练”，弹出图像分类和物体检测的对话框，选择需要的模型类型，如选择图像分类。

3.5.1.3 创建模型

进入模型中心，选择“创建模型”，根据个人需求填写模型的内容，带*号为必填选项。填写ok后选择“下一步”



3.5.1.4 训练模型

进入“训练模型”，选择模型类别，算法选择“通用”，训练方式选“默认”，因需要离线部署在设备上，**必须勾选“离线SDK”**，选择”添加训练数据“完成之后，勾选“增加识别结果为其他的默认分类”，选择”开始训练“。根据训练数据集的大小训练的时间会有些差异，一般100张左右的图片最快10分钟可训练完毕。

3.5.1.5 生成SDK

1.训练完成后，在我的模型中申请发布模型，选择软硬一体方案中的"EdgeBoard+专用SDK"，提交申请，等待审核

2.模型审核成功后，在”我的模型“处，点击”服务详情“按钮，在弹出的对话框中，选择下载SDK。

3、获取序列号

点击“管理序列号“跳转至百度云-->EasyDL定制训练平台-->离线SDK管理界面，查看用于激活sdk的序列号。

3.5.1.6 在EdgeBoard里安装SDK

1. 下载的软件部署包解压后，包含了简单易用的SDK和Demo。只需简单的几个步骤，即可快速部署运行EdgeBoard。部署包文件结构如下

```
EasyEdge-m1800-edgeboard/
├── cpp
|   └──
baidu_easyedge_linux_cpp_aarch64_PADDLEMOBILE_FPGA_v0.3.2_gcc6.2_20190518
|       ├── demo
|       |   ├── CMakeLists.txt
|       |   ├── demo.cpp
|       |   └── easyedge_serving
|       ├──include
|       |   ├── easyedge
|       |       └── easyedge.h
|       └── lib
|           ├── libeasyedge.so -> libeasyedge.so.0.4.0
|           ├── libeasyedge.so.0.4.0
|           ├── libeasyedge_static.a
|           ├── libpaddle-mobile.so
```

```
|           └── libverify.alib
├── RES
|   ├── conf.json
|   ├── label_list.txt
|   ├── model
|   ├── params
|   └── preprocess_args.json
|
└── tools
```

2. 使用序列号License 激活SDK

   打开demo.cpp文件

   【文件路径：EasyEdge-m1800-edgeboard/cpp/baidu_easyedge_linux_cpp_aarch64_PADDLEMOBILE_FPGA_v0.3.2_gcc6.2_20190518 /demo/demo.cpp】

   写入license序列号

   将set_licence_key函数中的字符串参数"set your license here"替换为序列号License即可

```
int main(int argc, char *argv[]) {

    if (argc != 3) {
        std::cerr << "Usage: demo {model_dir} {image_name}";
        exit(-1);
    }

    PaddleFluidConfig config;
    config.model_dir = argv[1];
    global_controller()->set_licence_key("set your license here");
    global_controller()->log_config.enable_debug = false;
    auto predictor = global_controller()->CreateEdgePredictor
<PaddleFluidConfig, EdgeEngineKind::kPaddleMobile>(
            config);
```

3. 将SDK（完全解压后）放到EdgeBoard系统/home/root/workspace/目录下（放入方法参见上文“文件拷贝”），然后按下述方法进行启动运行。

3.5.2 运行SDK

1.加载驱动,系统启动后加载一次即可（也可以加系统启动脚本）

```
insmod /home/root/workspace/driver/fpgadrv.ko
```

若未加载驱动，可能报错：

```
Failed to to fpga device: -1
```

设置系统时间（系统时间必须正确）

```
date --set "2019-5-18 20:48:00"
```

2.编译

```
//进入cpp文件的demo文件夹

cd /home/root/workspace/EasyEdge-m1800-
edgeboard/cpp/baidu_easyedge_linux_cpp_aarch64_PADDLEMOBILE_FPGA_v0.3.2_gcc6.2_2
0190518/demo

// 如果没有build目录，创建一个
mkdir build
cd build
rm -rf *
//调用cmake 创建 Makefile
cmake ..
// 编译工程
make
```

3.执行示例

```
//在build目录下运行执行文件
./easyedge_demo  {RES资源文件夹路径}  {测试图片路径}
```

例：在SDK中放入需要预测的图片，如将预测图片放入RES文件夹中，

```
 ./easyedge_demo /home/root/workspace/EasyEdge-m1800-edgeboard/RES/
/home/root/workspace/EasyEdge-m1800-edgeboard/RES/1.jpg
```

便可看到识别结果

3.5.3 HTTP服务调用

1.加载驱动,系统启动后加载一次即可（也可以加系统启动脚本）

```
insmod /home/root/workspace/driver/fpgadrv.ko
```

若未加载驱动，可能报错：

```
Failed to to fpga device: -1
```

设置系统时间（系统时间必须正确）

```
date --set "2019-5-18 20:48:00"
```

2.部署包中附带了HTTP服务功能，可直接运行

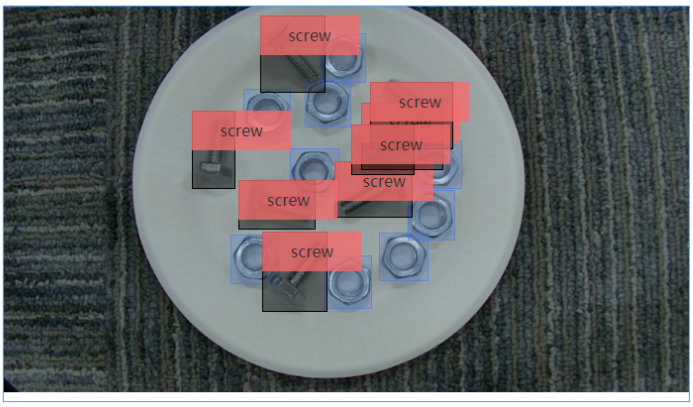
```
# ./easyedge_serving {RES目录} {序列号} {绑定的host，默认0.0.0.0} {绑定的端口，默认
24401}
cd ${SDK_ROOT}
export LD_LIBRARY_PATH=./lib
./demo/easyedge_serving /home/root/workspace/EasyEdge-m1800-edgeboard/RES/
 "1111-1111-1111-1111"
```

日志显示

```
2019-07-18 13:27:05,941 INFO [EasyEdge] [http_server.cpp:136] 547974369280
Serving at 0.0.0.0:24401
```

则启动成功。此时可直接在浏览器中输入`http://{EdgeBoard ip地址}:24401`，在h5中测试模型效果。

上传图片



结果

标签	置信度
nut	1.00
nut	1.00
nut	1.00
nut	0.99
screw	0.99
screw	0.98
screw	0.95
screw	0.92
screw	0.86
screw	0.85
screw	0.64
screw	0.34

3.5.4 HTTP 私有服务请求说明

- http 请求参数

URL中的get参数：

参数	说明	默认值
threshold	阈值过滤， 0~1	0.1

HTTP POST Body即为图片的二进制内容(无需base64, 无需json)

Python请求示例

```
import requests

with open('./1.jpg', 'rb') as f:
    img = f.read()
    result = requests.post(
        'http://127.0.0.1:24401/',
        params={'threshold': 0.1},
        data=img).json()
```

Java请求示例

- http 返回数据

字段	类型说明	其他
error_code	Number	0为成功,非0参考message获得具体错误信息
results	Array	内容为具体的识别结果。其中字段的具体含义请参考`预测图像-返回格式`一节
cost_ms	Number	预测耗时ms，不含网络交互时间

返回示例

```
{
    "cost_ms": 52,
    "error_code": 0,
    "results": [
        {
            "confidence": 0.94482421875,
            "index": 1,
            "label": "IronMan",
            "x1": 0.059185408055782318,
            "x2": 0.18795496225357056,
            "y1": 0.14762254059314728,
            "y2": 0.52510076761245728
        },
        {
            "confidence": 0.94091796875,
            "index": 1,
            "label": "IronMan",
            "x1": 0.79151463508605957,
            "x2": 0.92310667037963867,
            "y1": 0.045728668570518494,
            "y2": 0.42920106649398804
        },
      }
```

- 错误说明

SDK所有主动报出的错误，均覆盖在`EdgeStatus`枚举中。同时SDK会有详细的错误日志，开发者可以打开Debug日志查看额外说明：

```
global_controller()->log_config.enable_debug = true;
```

> 目前EdgeBoard暂不支持并行多模型计算。

4.进阶指南

4.1 开发应用

4.1.1 模型获得

目前Paddle-Mobile仅支持Paddle训练的模型。如果你手中的模型是不同种类的模型，需要进行模型转换才可以运行。验证过的网络包含resnet、Inception、ssd、mobilenet等。

- 训练模型:
  如果您没有模型，可以使用sample中的模型，或自己训练模型。
  1.通过PaddlePaddle开源深度学习框架自己训练模型，详细使用参考PaddlePaddle
  2.通过AI Studio平台训练模型，详细使用参考AI Studio
  3.可以在EasyDL等平台上传标注数据，训练模型，详细使用参考EasyDL
- 转换模型:
  1.如果您已有caffe模型，我们提供了相应的转换工具，帮助转为Paddle模型。详细使用参考X2Paddle_caffe2fluid
  2.如果您已有Tensorflow模型，我们提供了相应的转换工具，帮助转为Paddle模型。详细使用参考X2Paddle_tensorflow2fluid

4.1.2 连接视频数据源

EdgeBoard提供多种视频输入硬件接口，支持多种协议输入图像数据作为数据源。包括bt1120、usb、mipi等协议。

1. usb协议视频数据输入
   可以选择uvc usb摄像头作为视频源。插入usb摄像头到EdgeBoard接口4-USB3.0
2. bt1120协议视频数据输入
   可以选择海思具有bt1120视频数据输出的网络摄像头，通过fpc排线连接EdgeBoard接口14-BT1120，具体针脚的定义可参考硬件说明。
3. mipi协议视频数据输入
   可以选择适配好的mipi摄像头作为视频源，通过fpc连接EdgeBoard接口18-MIPI CSI。

4.1.3 加载驱动

使用EdgeBoard的加速功能，预测库会把计算量大的op通过驱动调用fpga进行运算。运行自己的应用前需要加载驱动，编译好的驱动位于/home/root/workspace/driver目录，提供无日志输出和有日志输出两个版本。
加载模型(若不想每次启动系统执行该命令，可以把加载驱动放到系统启动脚本中,参见常见问题与解答19)

```
insmod /home/root/workspace/driver/fpgadrv.ko
```

卸载驱动(正常情况您不需要卸载驱动，若需要加载有日志输出的版本，可以通过如下命令卸载后，再加载该版本）

```
rmmod /home/root/workspace/driver/fpgadrv.ko
```

4.1.4 使用预测库

EdgeBoard支持Paddle-Moblie预测库，编译好的预测库，位于/home/root/workspace/paddle-mobile。具体使用把预测库的头文件和动态库拷贝到自己应用中即可。另外可以参考我们提供的sample。Paddle-Moblie源码可以参考https://github.com/PaddlePaddle/paddle-mobile

4.1.5 创建应用

4.1.5.1 添加预测库

拷贝/home/root/workspace/paddle-mobilie/下面的动态库和头文件到您的工程中。在CmakeLists.txt添加paddle-mobile库的引用

```
set(PADDLE_LIB_DIR "${PROJECT_SOURCE_DIR}/lib" )
set(PADDLE_INCLUDE_DIR "${PROJECT_SOURCE_DIR}/include/paddle-mobile/" )

include_directories(${PADDLE_INCLUDE_DIR})
LINK_DIRECTORIES(${PADDLE_LIB_DIR})
...
target_link_libraries(${APP_NAME} paddle-mobile)
```

4.1.5.2 添加模型

拷贝自己训练的模型到您的工程中

4.1.5.3.添加预测数据源

可以选择图片，摄像头数据作为预测数据源，使用摄像头需要插入相应的摄像头。

- USB摄像头
  1) 插入摄像头后，通过ls /dev/video* 查看设备接入情况。通过会显示如下：

```
 /dev/video0  /dev/video1 /dev/video2
```

/dev/video2为usb摄像头v4l2输出yuv数据，当应用提示找不到设备时，可以修改src/video_classify.cpp或者src/video_detection.cpp。 通过/home/root/workspace/tools下video工具检测摄像头联通性

```
// src/video_classify.cpp 169行
config.dev_name = "/dev/video2";
```

2) 另外可以修改摄像头分辨率

```
// src/video_classify.cpp 170行
   config.width = 1280;
   config.height = 720;
```

3) 运行video工具

```
//读取usb摄像头，采集一张图片保存到本地
   cd /home/root/workspace/tools/video
   ./v4l2demo -i /dev/video2 -j -n 1
//如有疑问，查看帮助
   ./v4l2demo -h
```

执行程序后在build目录下会生成jpg文件，可以查看图片否正确。如果没有生成图片，检测是否识别到USB设备。

- bt1120 ipc摄像头
  Edgeboard通过bt1120协议接收原始数据进行推理后，可以把结果通过串口或spi传回ipc（bt1120、串口、spi接口定义参考硬件说明）。可以在图片帧的像素数据中携带帧编号。

  插入摄像头后，通过/home/root/workspace/tools下video工具检测摄像头联通性
  1) 查看设备, 正常情况设备为/dev/video1

```
  ls /dev/video*
  /dev/video0  /dev/video1
```

2) 配置摄像头参数

```
  media-ctl -v --set-format '"a0010000.v_tpg":0 [RBG24 1920x1080 field:none]'
```

3) 运行video工具

```
//读取BT1120摄像头，采集一张图片保存到本地
   cd /home/root/workspace/tools/video
   ./v4l2demo -i /dev/video1 -j -n 1
//如有疑问，查看帮助
   ./v4l2demo -h
```

执行程序后在build目录下会生成jpg文件，可以查看图片否正确。如果没有生成图片，检测bt1120连接线是否正确。

4.1.5.4 调用预测库加载模型和使用预测数据

- 初始化模型

```
 Predictor _predictor_handle = new Predictor();
 _predictor_handle->init(model, {batchNum, channel, input_height, input_width},
output_names);
```

- 准备数据
  1.缩放图片到指定的大小。如果网络只能固定大小输入，需要缩放到网络输入大小。
  2.图片预处理（减均值、转浮点、归一化等）.
  3.产出数据，由于EdgeBoard使用的NHWC格式，通常视频过来的数据就是NHWC格式，就不需要NHWC->NCHW转换。
- 预测数据
  调用API的predict接口，传输处理好的数据，获取预测结果

```
bool predict(const float* inputs, vector<float*> &outputs,vector<vector<int> >
&output_shapes);
```

4.2 驱动说明

驱动模块包含设备管理、内存管理、IO设定、参数管理及命令控制等功能
1.设备管理

```
int open_device();
void close_device();
void reset_device();
```

2.内存管理
驱动部分会为自己及FPGA设备从系统内存中保留自己专用的内存，这部分内存操作系统Linux在没有Driver的情况下看不到或不可独立操作，需要借助驱动的功能来使用，来完成相应内存管理（分配、释放、映射、拷贝等功能）

```
void* fpga_malloc(size_t size);
void fpga_free(void* ptr);
void fpga_copy(void* dst, const void* src, size_t num);
int fpga_flush(void* address, size_t size);
int fpga_invalidate(void* address, size_t size);
```

3.IO设定、参数管理及命令控制
同时驱动还为上层（顶层的主控应用）提供IO设定、参数管理、及命令控制等功能，并完成相应的、和FPGA设备的通信设定。

```
int PerformBypass(const struct BypassArgs& args);
int ComputeFpgaConv(const struct ConvArgs& args);
int ComputeFpgaPool(const struct PoolingArgs& args);
int ComputeFpgaEWAdd(const struct EWAddArgs& args);
int ComputeFPGAConcat(const struct ConcatArgs& args);
int ComputeScale(const struct ScaleArgs& args);
int ComputeNormalize(const struct NormalizeArgs& args);
int ConfigPowerParameter(const struct PowerParameterArgs& args);
int ConfigNormalizeParameter(const struct NormalizeParameterArgs& args);
int ConfigInplace(const struct InplaceArgs& args);
```

4.3 预测库说明

1.Paddle-Mobile
Paddle-Mobile是PaddlePaddle组织下的项目，是一个致力于嵌入式平台的深度学习预测框架。EdgeBoard使用的是Paddle-Mobile下的FPGA实现的预测库。

2.NHWC
基于FPGA的特性，Paddle-Mobile的FPGA实现数据格式为NHWC。在开发自己的应用或修改paddle-mobile代码时，需要注意。

```
void convert_to_hwc(float **data_in, int channel, int height, int width)
```

3.FP16(实现为half)
FPGA实现的OP,Tensor数据为FP16。当遇到CPU实现的OP时，需要把FP16转换层FP32(即Float)。反之，需要把FP32转换成FP16。

```
// FP16->FP32
float* float_data = (float*)fpga::fpga_malloc(height * cw_alinged *
sizeof(float));
fpga::to_float(const_cast<float*>(input_half->data<float>()), float_data, height
* cw_alinged);

// FP32->FP16
fpga::to_half(output_boxes_dataptr, boxes_data, output_boxes->numel());
```

4.对齐
基于FPGA的特性，使用FPGA实现的OP前需要将Tensor的数据基于C*W进行16位倍数进行对齐。反之，上一个节点是FPGA实现，下一个节点为CPU实现时需要反对齐

```
void align_element(char** data_in, int num, int chw);
void align_num(char** data_in, int num_per_div_before_alignment, int num,int
chw);
```

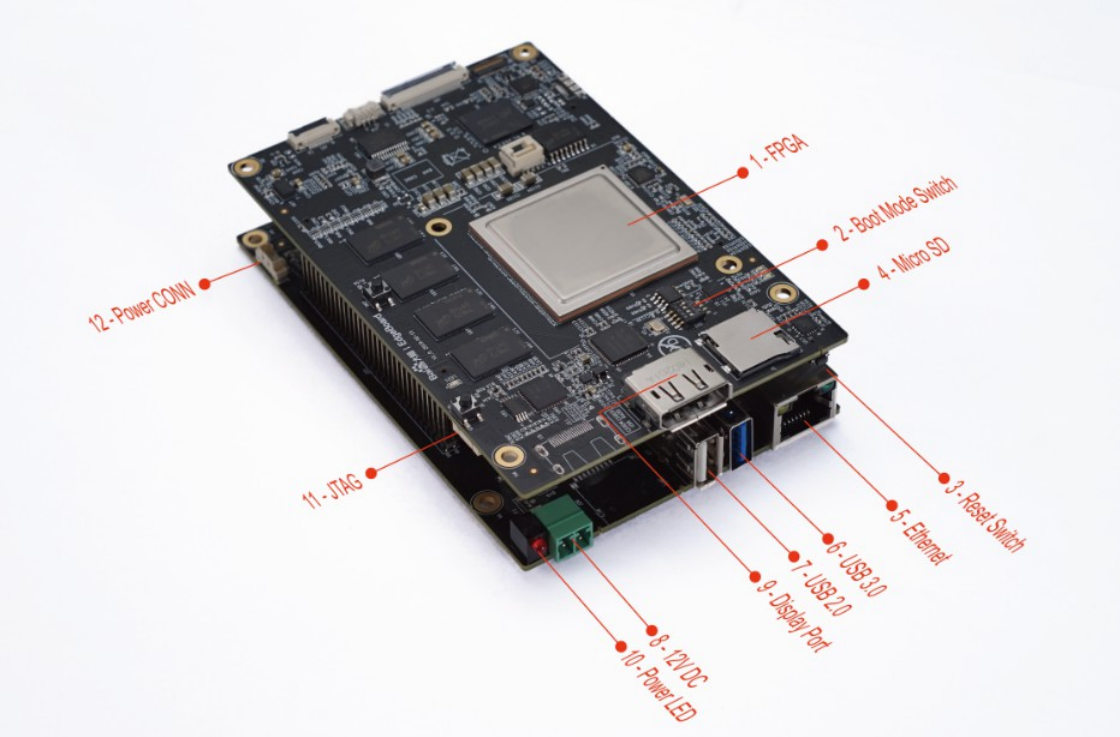
5.OP定制
当Paddle-Mobile现有的OP,无法满足您的模型需求时，可以新增或定制OP。具体可以参考[Paddle-Mobile OP代码设计]。OP定制需要主要NHWC、FP16、对齐等问题。

4.4 硬件说明

EdgeBoard高性能计算板卡使用的是Xilinx Zynq UltraScale+ MPSoCs 系列的芯片， Zynq芯片可分成处理器系统部分PS(Processor System)和可编程逻辑部分PL(Programmable Logic)。在EdgeBoard高性能计算板卡上搭载了丰富的外部接口和设备，方便用户的使用和功能验证。

4.4.1 主板结构与接口功能介绍

- EdgeBoard开发板的接口示意图



1-1 EdgeBoard开发板配置及接口功能介绍：

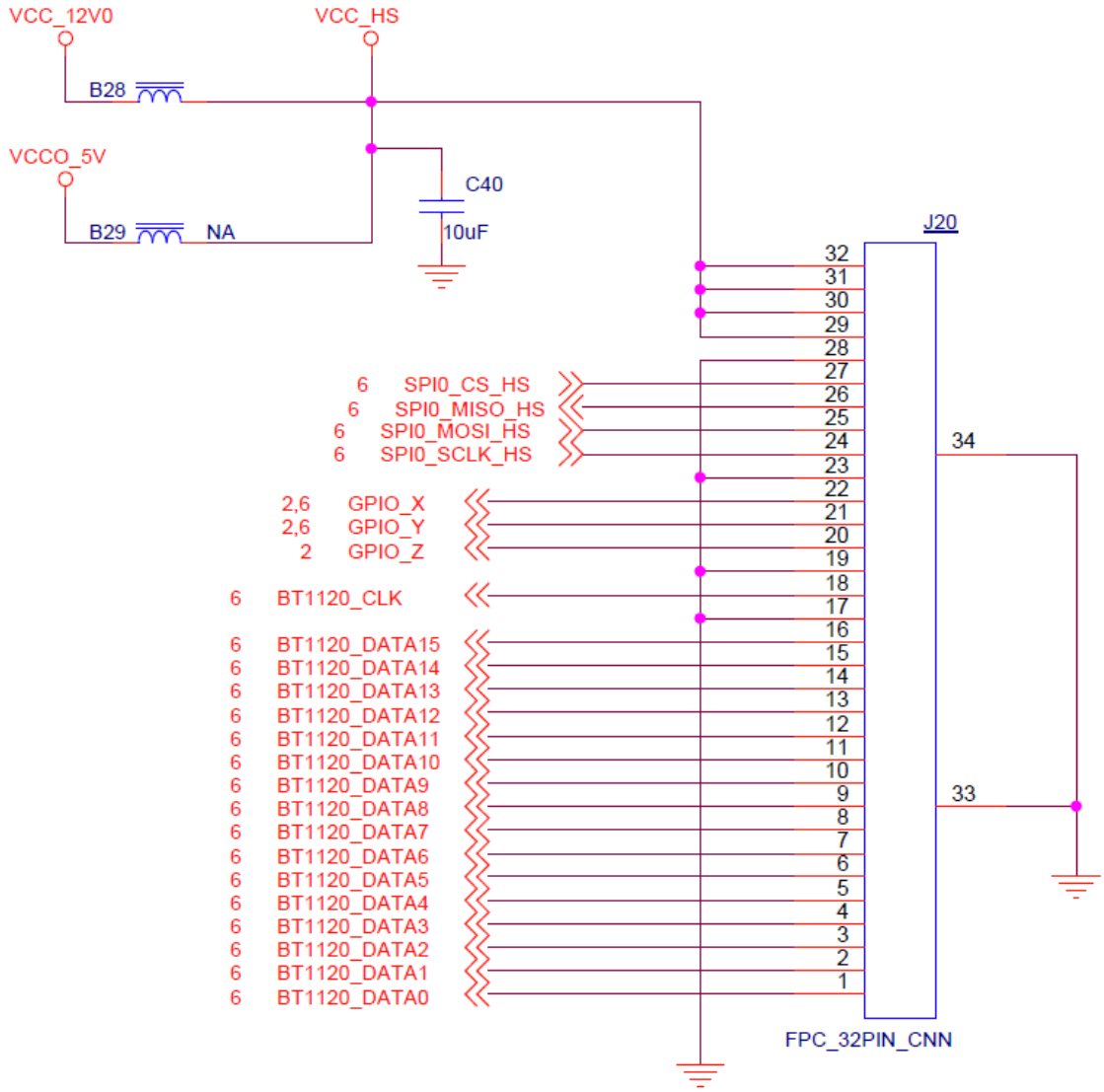
序号	芯片或接口	说明
1	MPSoC	Zynq UltraScale + MPSoCs ZU9EG Quad Core Cortex-A53@1.5GHz
存储设备		
13	DDR4	DDR4，2GB
20	Flash	QSPI Flash，256MB
19	eMMC	eMMC，8GB
4	MicroSD Card slot	16GB，最大支持128GB，用于存储操作系统镜像和文件系统，或者进行信息数据的存储等
视频输入接口		
16	BT1120 Camera	BT1120 video input , support 1080p
18	MIPI CSI	MIPI CSI video input,
6/7	USB	suppoer USB camera video input (UVC)
视频输出接口		
9	Display Port	视频输出接口，方便视频及图片直接显示
网口		
5	Ethernet	1000M以太网接口，进行以太网数据交互
调试接口		
11	JTAG	PL JTAG debug interface
17	USB UART	PS UART debug interface
预留扩展接口		
15	UART	FPGA UART 接口，预留与外部设备通信
16	USB&SDIO	FPGA USB和SDIO的扩展口，预留与外部设备通信
电源		
8	12V DC	12V DC Power Supply Interface
17	Power CNN	12V DC Power Supply CONN，预留与外部设备供电
LED灯		
9	Power LED	Red LED，12V DC Power LED

序号	芯片或接口	说明
按键		
2	Boot Mode Switch	用来调节开发板启动方式
3	Reset Switch	系统复位按键

4.4.1.1 Main-Board主板端BT1120接口pin定义

Pin number	Pin name	Pin number	Pin name
32	VCC_5V/VCC_12V0	16	BT1120_DATA15
31	VCC_5V/VCC_12V0	15	BT1120_DATA14
30	VCC_5V/VCC_12V0	14	BT1120_DATA13
29	VCC_5V/VCC_12V0	13	BT1120_DATA12
28	GND	12	BT1120_DATA11
27	SPI0_CS_HS	11	BT1120_DATA10
26	SPI0_MISO_HS	10	BT1120_DATA9
25	SPI0_MOSI_HS	9	BT1120_DATA8
24	SPI0_SCLK_HS	8	BT1120_DATA7
23	GND	7	BT1120_DATA6
22	GPIO_X	6	BT1120_DATA5
21	GPIO_Y	5	BT1120_DATA4
20	GPIO_Z	4	BT1120_DATA3
19	GND	3	BT1120_DATA2
18	BT1120_CLK	2	BT1120_DATA1
17	GND	1	BT1120_DATA0

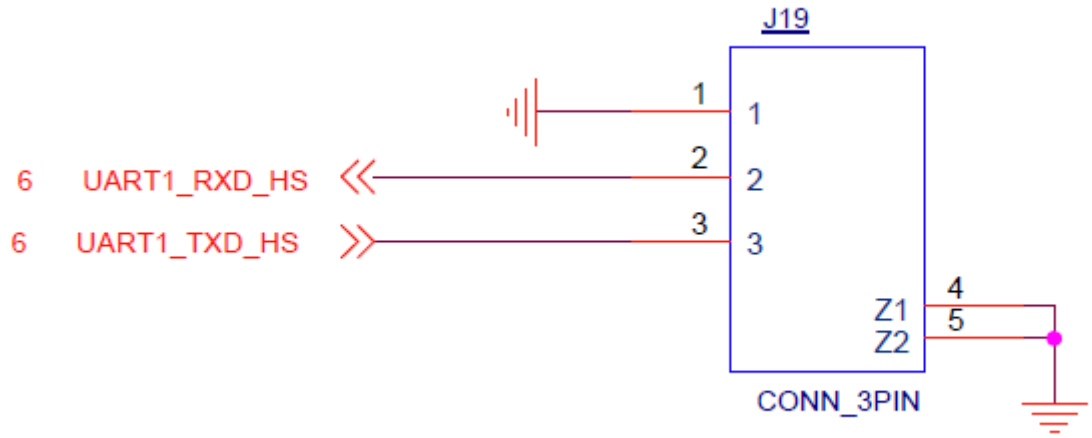
BT1120接口J20为32pin掀盖式FPC连接器，pin pitch 0.5mm，在主板端BT1120的data & clk 信号的电平为3.3V，所以需要外部接入的BT1120信号电平为3.3V。BT1120接口原理图如下：



4.4.1.2 Main-Board主板端UART接口pin定义

Pin number	Pin name
1	GND
2	UART1_RXD_HS
3	UART1_TXD_HS

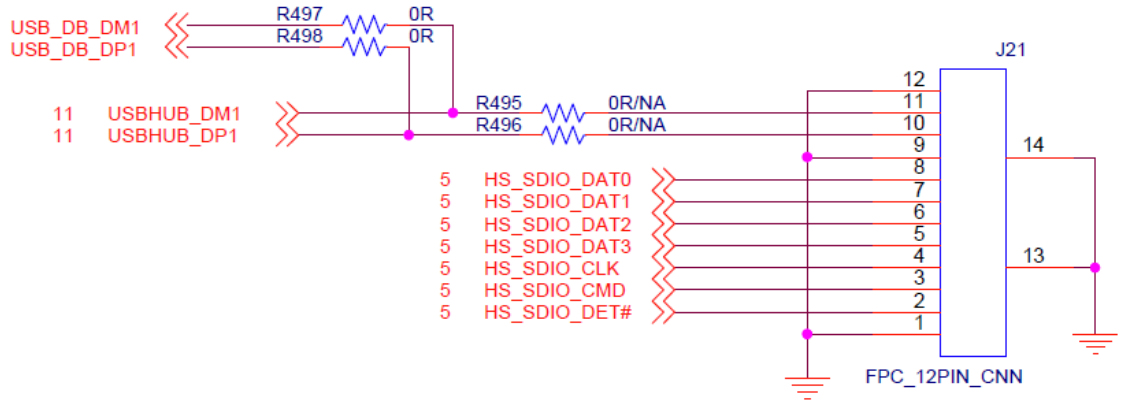
UART接口J19为3pin直插式连接器，pin pitch为1.25mm，在主板端UART 信号的电平为3.3V，所以需要外部接入的UART信号电平为3.3V。UART接口原理图如下：



4.4.1.3 Main-Board主板端SDIO&USB接口pin定义

Pin number	Pin name	Pin number	Pin name
1	GND	7	HS_SDIO_DAT1
2	HS_SDIO_DET#	8	HS_SDIO_DAT0
3	HS_SDIO_CMD	9	GND
4	HS_SDIO_CLK	10	USBHUB_DP1
5	HS_SDIO_DAT3	11	USBHUB_DM1
6	HS_SDIO_DAT2	12	GND

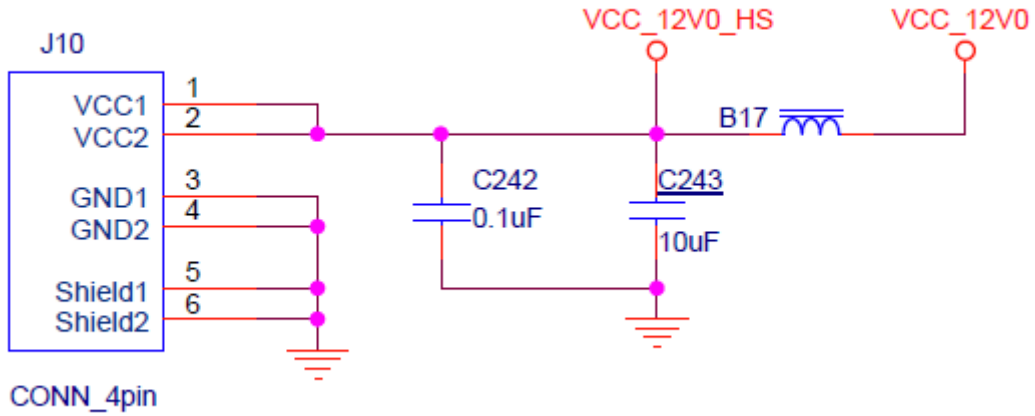
SDIO&USB接口J21为12pin掀盖式FPC连接器，pin pitch为0.5mm，在主板端SDIO信号电平为1.8V，USB信号为3.3V，所以需要外部接入的SDIO信号电平为1.8V，USB信号电平为3.3V。SDIO&USB接口原理图如下：



4.4.1.4 Power-Board主板端12V Power接口pin定义

Pin number	Pin name	Pin number	Pin name
1	VCC_12V0	3	GND
2	VCC_12V0	4	GND

12V Power接口J10为4pin直插式连接器，pin pitch为1.25mm，12V Power接口原理图如下：

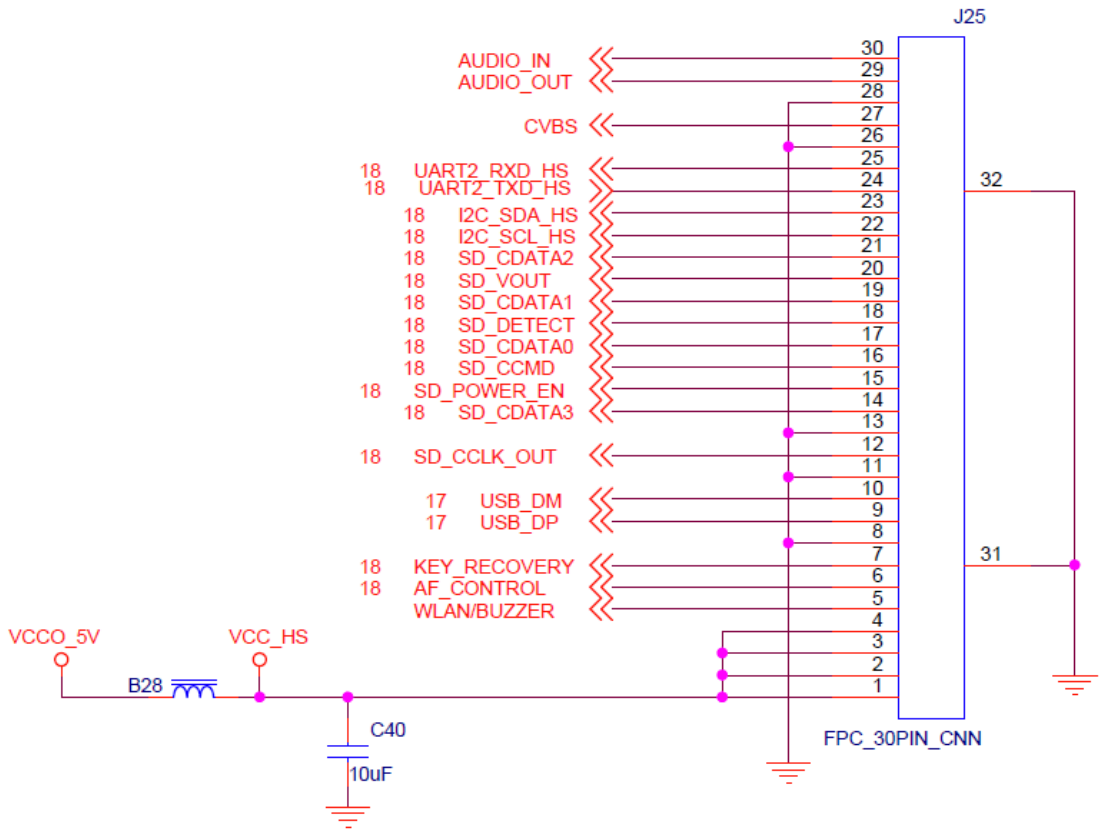


4.4.1.5 Power-Board主板端SDCard & USB2.0 扩展接口pin定义：

Pin number	Pin name	Pin number	Pin name
30	-	15	SD_POWER_EN
19	-	14	SD_DATA3
28	GND	13	GND
27	-	12	SD_CCLK_OUT
26	GND	11	GND
25	UART2_RXD_HS	10	USB_DN
24	UART2_TXD_HS	9	USB_DP
23	I2C_SDA	8	GND

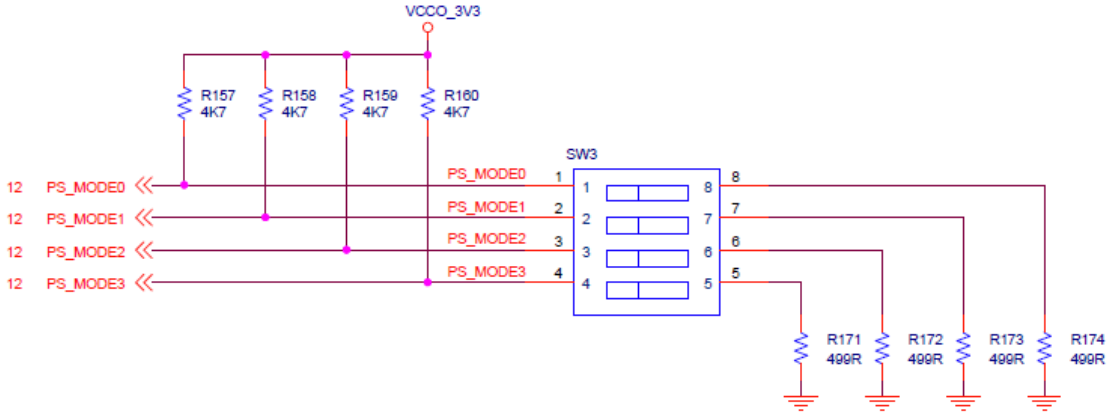
Pin number	Pin name	Pin number	Pin name
22	I2C_SCL	7	KEY_RECOVERY
21	SD_CDATA2	6	AF_CONTROL
20	SD_VOUT	5	-
19	SD_CDATA1	4	VCC_5V
18	SD_DETECT	3	VCC_5V
17	SD_CDATA0	2	VCC_5V
16	SD_CCMD	1	VCC_5V

SDCard & USB2.0 扩展接口J25为30pin掀盖式FPC连接器，pin pitch 0.5mm，接口原理图如下：



4.4.2 启动模式配置

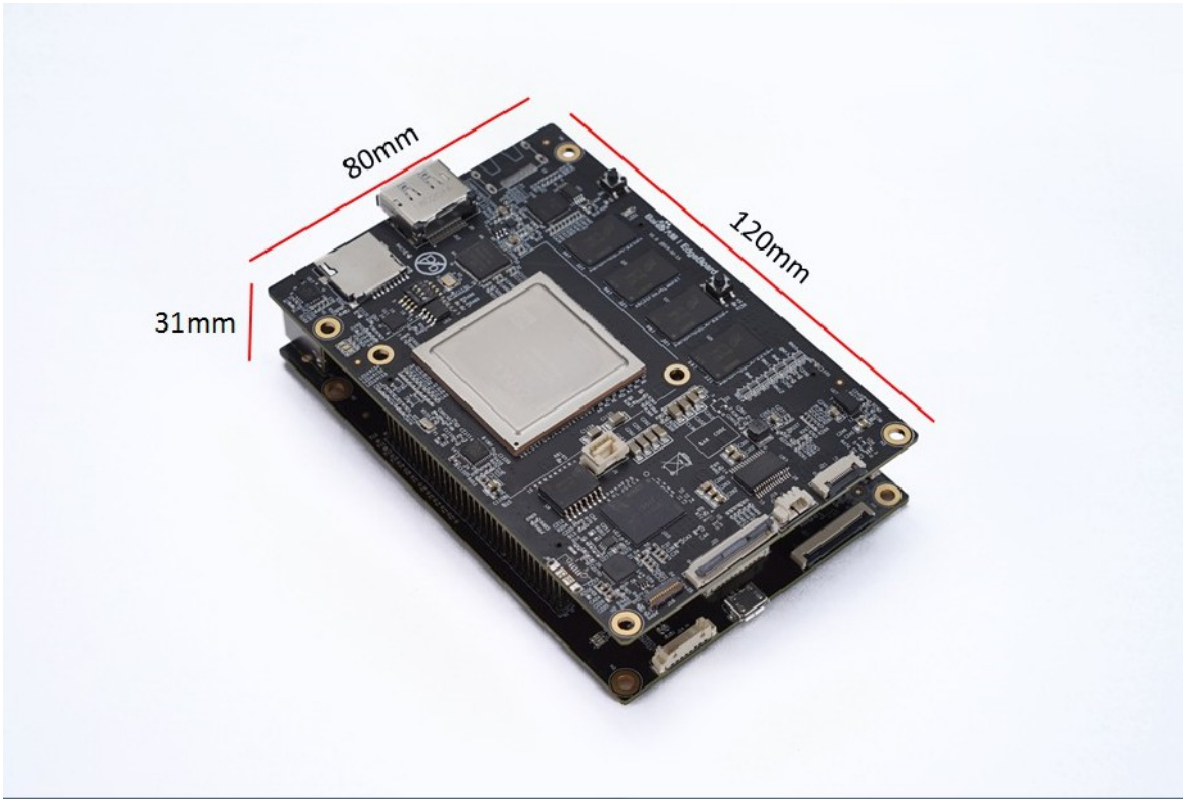
EdgeBoard开发平台支持四种启动模式。这四种启动模式分别是JTAG调试模式,QSPI FLASH启动模式、eMMC启动模式、SD卡启动模式。FPGA芯片上电后会检测响应MIO口的电平来决定那种启动模式。用户可以通过核心板上的拨码开关SW3来选择不同的启动模式。SW3启动模式配置如下表所示。



BOOT MODE	SW3 [4-1]
JTAG	ON-ON-ON-ON
QSPI24	ON-ON-ON-OFF
QSPI32	ON-ON-OFF-ON
eMMC	ON-OFF-OFF-ON
SD Card	OFF-OFF-OFF-ON

4.4.3 EdgeBoard结构尺寸

EdgeBoard板卡尺寸为120mmx80mmx31mm，PCB采用多层板设计。板子四周有4个螺丝定位孔，可以用于固定板卡，定位孔的孔径为3.5mm(直径)，如下图：



5. 常见问题与解答

1. EdgeBoard是什么？它有什么作用？

EdgeBoard 是基于FPGA打造的嵌入式AI解决方案及基于此方案实现的系列硬件，与EasyDL等模型定制平台深度打通，具有高性能、高通用、易开发等三大优点，适用于开发验证、产品集成、科研教学、项目落地等应用方向，以及安防监控、工业质检、医疗诊断、农作物生长监控、无人驾驶、无人零售等应用场景。EdgeBoard基于linux系统，整个开发过程就是一个linux应用程序的开发。应用程序获取视频输入，调用预测库加载模型，调度模型，驱动加速模块进行计算，加速模型运行，获得运行结果。

2.EdgeBoard的优势

良好兼容百度大脑丰富的预置模型，及定制化模型，支持主流深度学习框架转换。 高性能表现，计算性能实测高于终端CPU计算卡50倍。具备丰富的开发工具与接口，让开发简单轻便。

3. EdgeBoard都提供哪些东西

EdgeBoard开发套件除了硬件板卡，还提供了完成开发工具链，包含了带有深度学习加速功能的定制Linux操作系统，二次开发环境，预测库PaddleMobile，模型的转化工具及一些示例工程。

4. 除了开发板有其它配件吗

EdgeBoard目前专注于图像推理，视频输入（参见2.2.4）可以配合USB摄像头、配套的mipi摄像头和海思网络摄像头等使用，视频输出（参见2.2.3）搭配DP显示器或者使用转换器转换成其他类型的显示器也可使用，陆续我们会与合作伙伴推出更多配件。

5.能用来训练模型吗？

目前EdgeBoard专注于模型推理，暂不支持模型训练。模型训练可使用EasyDL平台与Paddle框架。

6.开发前需要准备哪些东西？

EdgeBoard预置了一些示例工程，开发者可直接学习调试。开发者在正式进行业务开发时，需要准备好适用于业务场景的模型。

7.都持哪些框架？我的Caffe模型的能用吗？

目前主要支持PaddlePaddle框架模型，TensorFlow/Caffe需要用我们提供的工具进行转换，即可正常使用。

8.我不会训练模型，怎么办？

推荐购买百度已开放的模型算法，或者使用百度EasyDL平台，支持零开发基础定制模型，详见：http://ai.baidu.com/easydl/

9.能跑应用程序吗?用什么语言开发？

我们的芯片将强大的实时处理器与可编程逻辑集成在一起，可以看作是传统的FPGA和ARM集合而成的SoC。我们在EdgeBoard上预置了Linux操作系统，所以开发流程就是标准的Linux软件开发流程。目前支持C++的开发，后续会提供Python接口，开发更简单。

10.都支持哪些神经网络？

目前我们支持SSD,VGG,Resnet,Mobilenet,FaceBox等经典神经网络，未来将会验证及支持更多的神经网络。

11.都支持哪些OP呢？

目前支持的OP可以参考: https://github.com/PaddlePaddle/paddle-mobile (Github,Paddle-Mobile仓库)，我们也在持续迭代支持更多的OP。

12.我能自己定义扩展OP吗？

我们现在是基于paddle-mobile开发的，paddle-mobile本身是开源代码。对于非密集型的OP，您可以自己使用C++实现。对于运算量非常大，CPU实现拥有性能瓶颈的OP，您可以与我们联系（EdgeBoard QQ群：686301734），我们来做定制化开发。

13.都支持哪些接口？

图像输入有BT1120（用于海思网络摄像头）,并口相机,MIPI（用于MIPI摄像头）,USB（用于USB摄像头）和以太网口等高速输入口。还有SPI,Uart通用低速接口与其它芯片通信。 开发板上也有DP接口用于视频输出。 具体详见硬件接口介绍。

14.我要的接口不支持怎么办？

针对我们不支持的接口，可以考虑是否能通过中间转接来实现，或者修改我们的底板加入自己的接口。

15.能当通用的FPGA开发板吗？

目前暂不支持

16.我不懂FPGA，我能用吗？

EdgeBoard正是面向不懂FPGA开发和深度学习的用户，由百度完成FPGA逻辑设计，驱动设计并封装底层的功能。封装深度学习相关内容，用户只需调用paddle-mobile API接口，编写少量的业务代码即可完成神经网络在终端设备上的运行。从模型生产、配套硬件、软件开发到实际场景运用，全链路支持。

17.初次启动

如果使用串口调试，首先保证host电脑已经安装的相应的设备驱动和调试工具（参见2.1调试设备），如果使用网口调试，EdgeBoard支持ssh、samba、ftp网络服务，网口调试不需要安装驱动，直接使用调试工具打开ssh服务，配置host电脑或者路由器改为和EdgeBoard同一网段，输入EdgeBoard默认的静态ip:192.168.1.254，以及登录名密码root/root，即可登录系统（参见2.1.1）。EdgeBoard提供了完整的深度学习加速套件以及丰富的神经网络模型示例，上电即可体验。

18.自定义模型加载流程是什么样的？

1 ）创建工程，添加预测库
2 ）添加模型（自训练的模型或转换的模型）
3 ）添加预测数据源（图片、摄像头）
4 ）调用预测库加载模型和使用预测数据
5 ）运行调试
以上步骤可以参考EdgeBoard自带的示例工程

19.每次跑模型都要加载驱动，驱动怎么能自动加载

1）在系统中添加自启动脚本

```
// 打开启动目录
cd /etc/init.d/
// 新建启动脚本并编辑，名称可以自定义
vim eb.sh
```

脚本内容

```
chmod +x /home/root/workspace/driver/fpgadrv.ko
insmod /home/root/workspace/driver/fpgadrv.ko
```

2）建立软链接

```
cd /etc/rc5.d/
ln -s /etc/init.d/eb.sh S99eb
```

3）更改脚本权限

```
chmod +x /etc/init.d/eb.sh
reboot
```

20.软件更新说明

我们会在2周或4周进行一次更新，版本更新说明及相应的软件更新包会在文档中的版本目录。更新主要包含：

1 ）sample更新
连接samba或ftp，拷贝sample到/home/root/workspace/
2 ）driver更新
连接samba或ftp，拷贝driver到/home/root/workspace/
3 ）paddle_mobile预测库更新
连接samba或ftp，拷贝paddle_mobile到/home/root/workspace/paddle_mobile
4 ）系统镜像文件（包含fpga相关更新）BOOT.BIN和image.ub更新
a、连接samba或ftp，拷贝BOOT.BIN和image.ub到/home/root/workspace/
mount /dev/mmcblk1p1 /mnt/sdcard/
cp /home/root/workspace/BOOT.BIN /mnt/sdcard/
cp /home/root/workspace/image.ub /mnt/sdcard/
reboot
b、将SD卡通过读卡器插入电脑，使用新的BOOT.BIN和image.ub替换小分区中已有的BOOT.BIN和image.ub。
samba和ftp的使用请参照2.1.2文件拷贝。

21.我们的联系方式

1）EdgeBoard交流论坛

http://ai.baidu.com/forum/topic/list/198

2）EdgeBoardQQ群

群 号：686301734

3）官网地址

https://ai.baidu.com/tech/hardware/deepkit

4) 产品经理微信