RK1108 DSP 系统介绍

Sep., 2016

余智超

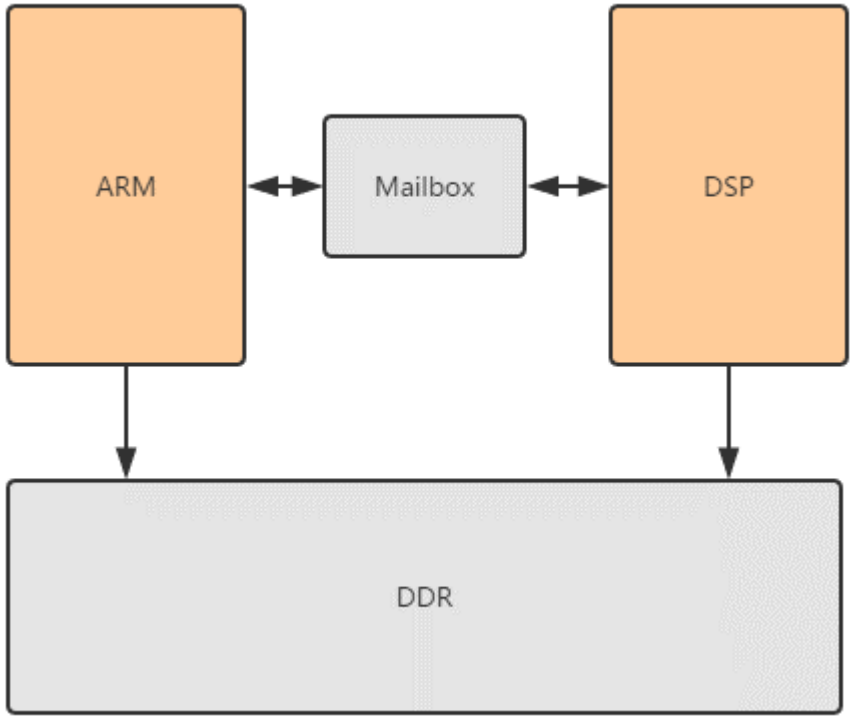
Agenda

- 简介
- DSP 软件基本框架
- 框架层 DPP
- DSP 驱动介绍
- DSP 系统介绍
- DSP 开发说明
- Q&A

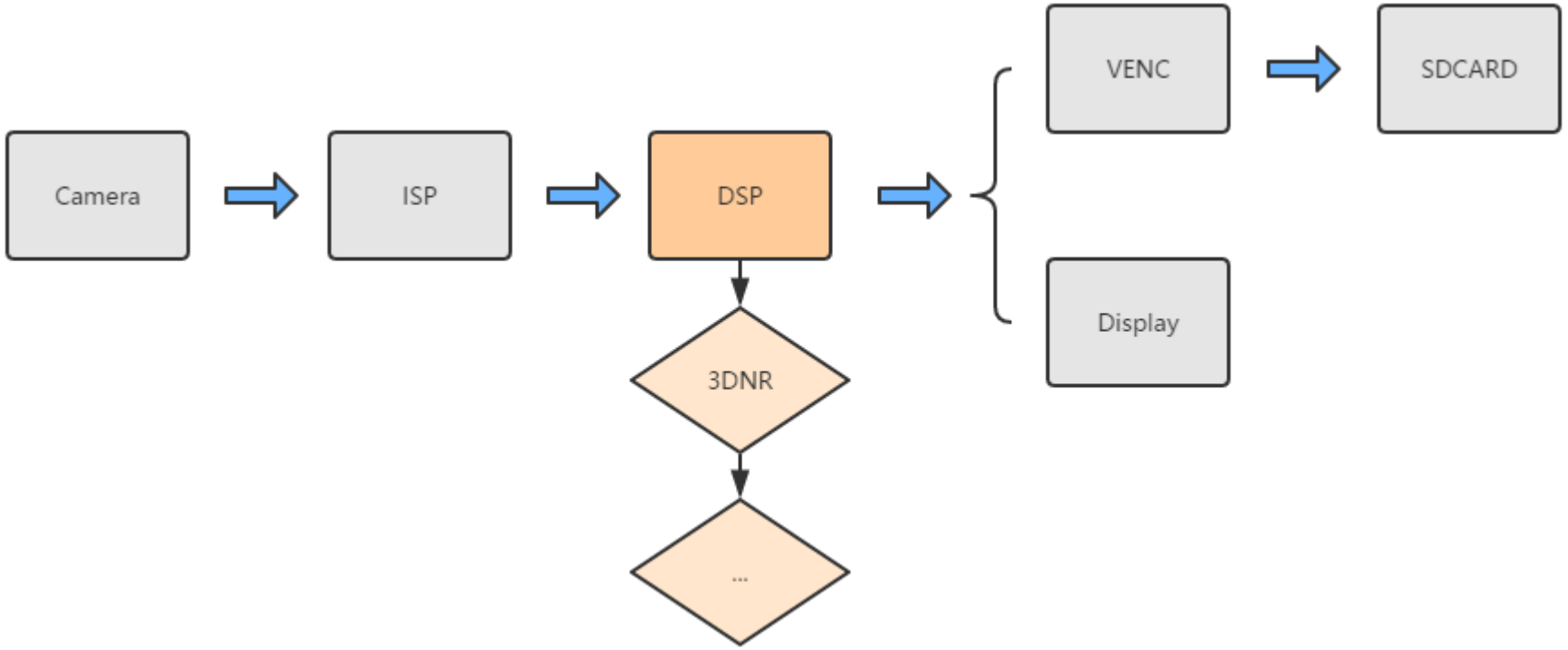
简介

RK1108 SoC DSP Architecture

- DSP 为 RK1108 的主要运算单元
- DSP 可访问外部 DDR ，应用范围广泛
- DSP 和 ARM 之间通过 Mailbox 进行通讯
- DSP 运算能力强大，未优化前 800MS 的计算量，经过优化后只需要 20MS，还可以更低
- 可根据算法的复杂度选择合适的 DSP 频率，从而达到能耗最优化



DSP 在 CVR 上的应用



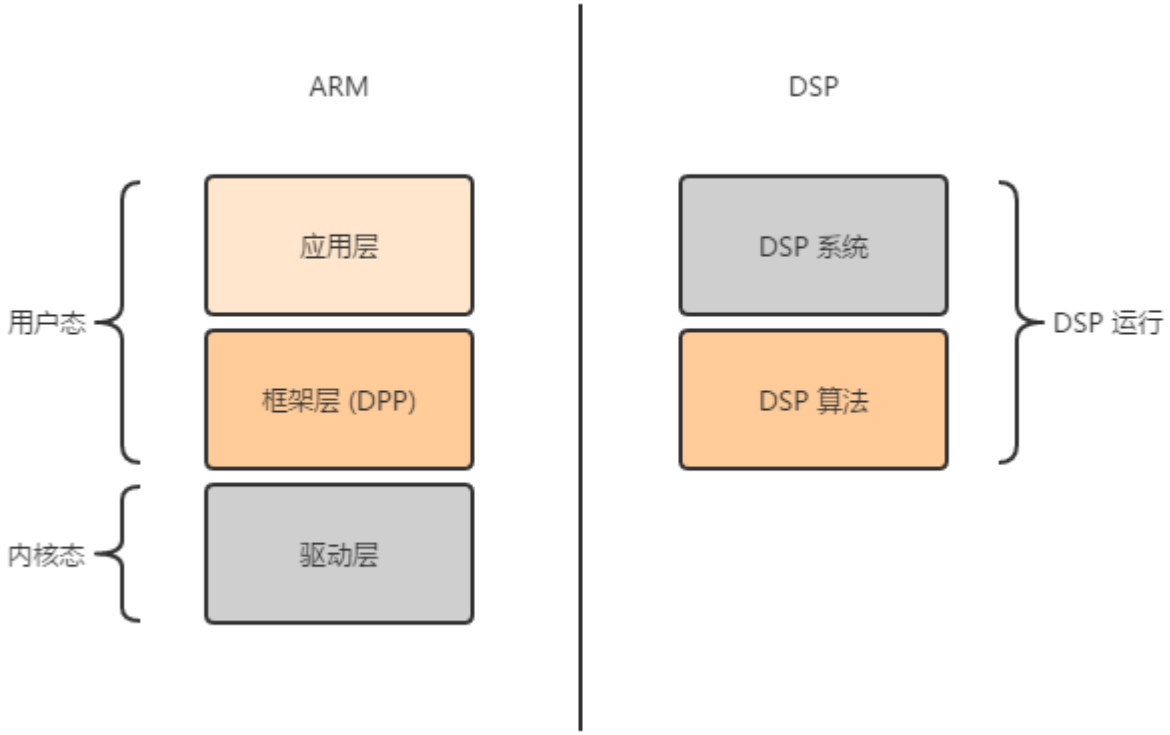
通过 3DNR 来降低 ISP 输出图像的噪声

DSP软件框架介绍

RK1108 DSP Software Architecture

Case 1：如果客户不需要添加算法，则只需要开发应用层

Case 2：如果客户添需要加自己的算法，则需要开发应用层、框架层以及 DSP 算法



DSP Software Layer Introduction

1、应用层

<SDKRoot>/app/video，实现了 CVR 的功能，串联起各个模块，通过调用 DPP 接口来使用 3DNR 算法和ADAS算法处理 ISP 出来的图像数据

2、框架层（DPP）

<SDKRoot>/external/dpp，管理输入输出 Buffer，管理工作队列，准备好算法参数，调用 DSP Kernel Driver。目前 DPP 没有开源，已经支持了 3DNR 和 ADAS

3、DSP Kernel Driver

<SDKRoot>/kernel/arch/arm/mach-rockchip/dsp，负责引导和加载 DSP、ARM 和 DSP 的通讯、DSP 设备的电源状态以及管理各个进程的工作请求。

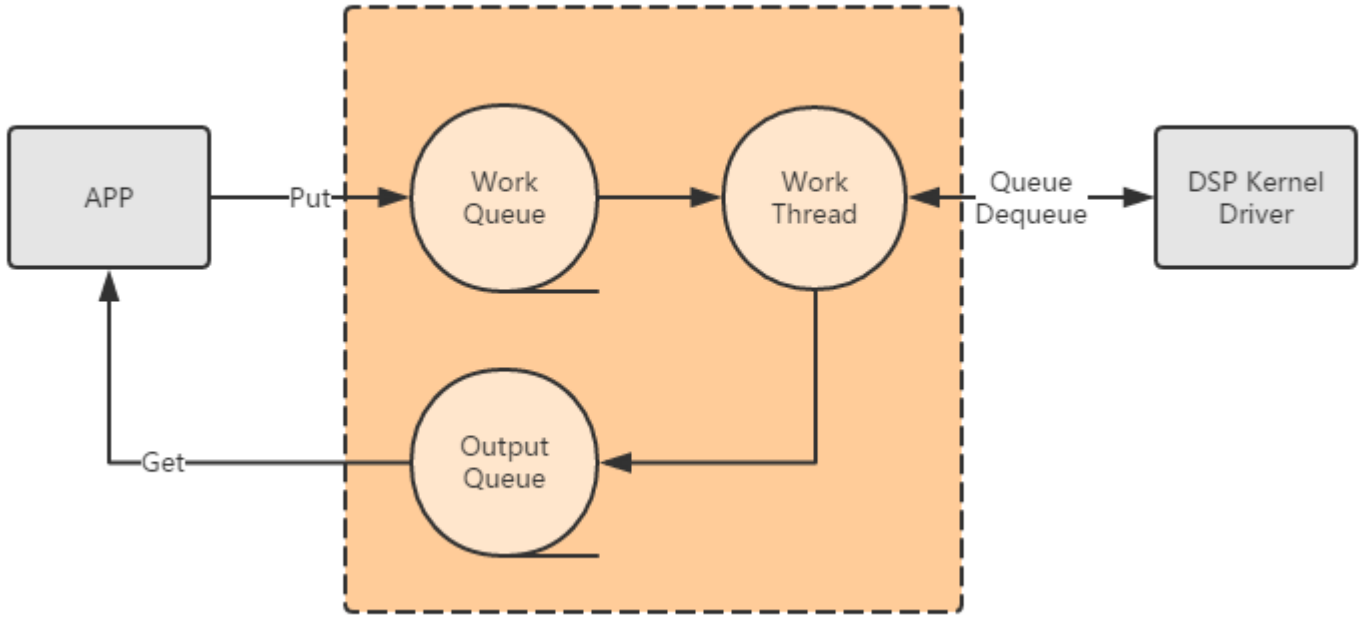
4、DSP System & DSP Algorithm

<SDKRoot>/external/dpp/out/firmware/rkdsp.bin，实际运行在DSP上的代码，负责监听ARM的工作请求，调用算法，并返回处理结果

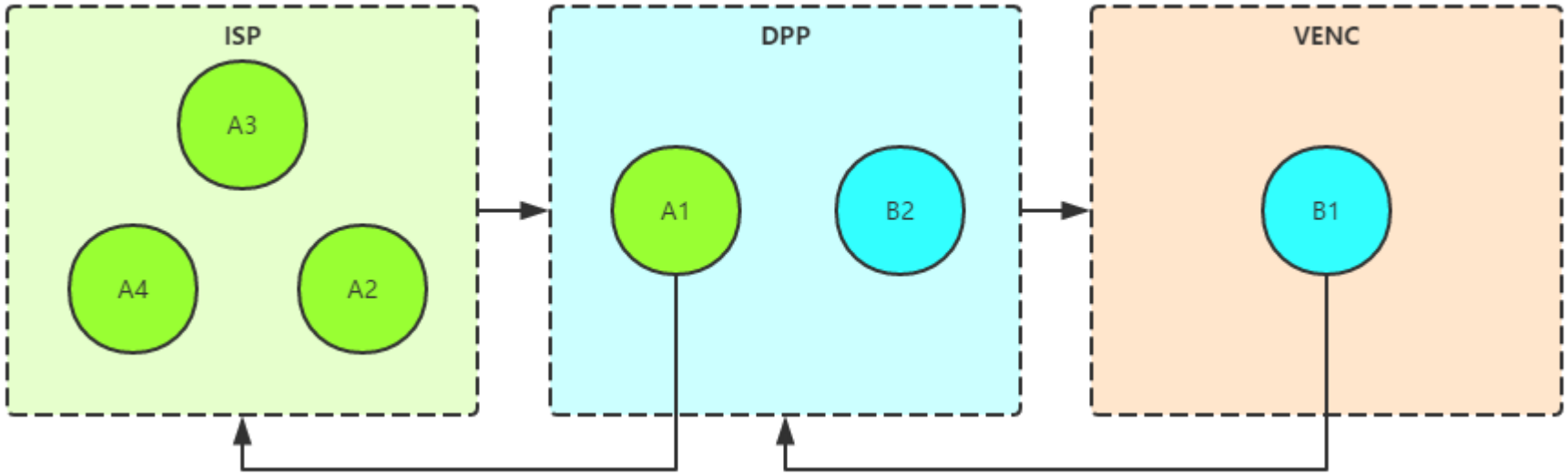
框架层 DPP

DSP Process Platform Introduction

- 管理输入输出Buffer
- 管理工作队列
- 准备 DSP 工作参数
- 向应用层返回 DSP 处理结果



DPP Buffer Management



- 需要减少 Buffer 的申请和拷贝
- 必须处理好 Buffer 引用关系
- DSP 操作的 Buffer 必须物理连续，应从 CMA 分配

DPP 接口说明

DPP 框架的接口头文件在 <SDKRoot>/external/dpp/inc 目录里面

1、dpp_create() – 创建 dpp context，一个 dpp context 是对一个算法的抽象。

2、dpp_start() – 启动 dpp 工作线程，dpp 开始处理工作队列

3、dpp_packet_init() – 创建 dpp packet，packet 中包含了算法所需的参数，由应用层配置

4、dpp_put_packet() – 向 dpp 工作队列添加工作请求包

5、dpp_get_frame() – 从 dpp 获取处理结果

6、dpp_stop() – 停止 dpp 工作线程

7、dpp_destroy() – 销毁 dpp context

DSP 驱动介绍

DSP Kernel Driver Introduction

DSP 驱动主要功能：

- 引导和启动 DSP
- 管理不同进程的 DSP 工作请求
- 管理 ARM 和 DSP之间的通讯
- 管理 DSP 的电源状态
- DSP 算法参数对 DSP 驱动来说是透明的，通用性强
- DSP 驱动是开源的，客户一般不需要改动

DSP Kernel Driver User Interface

- 设备文件 /dev/dsp

  用户态进程通过 open() 接口打开 DSP 设备文件来操作 DSP 驱动

- IOCTL

  用 ioctl() 接口调用DSP驱动的 IOCTL：

  1) DSP_IOC_QUEUE_WORK：

  传递工作参数 struct dsp_user_work 给 DSP，请求 DSP 调用适合的算法进行运算

  2) DSP_IOC_DEQUEUE_WORK

  获取 DSP 的计算结果，此接口为阻塞接口，会在 kernel 中等到 DSP 工作完成

DSP Kernel Driver Work Parameter

- packet 作为一个容器，客户可以将自己定制化的结构体作为 packet 传递给 DSP 驱动，驱动会将这些参数透传给 DSP
- DSP 操作的 Buffer 必须物理连续
- DSP 会根据 type 的值来选择算法运行

```
72 /*
73  * dsp_render_params - parameters used by DSP core
74  * hardware to render a frame
75  *
76  * @type: render type, DSP_RENDER_3DNR etc
77  * @reserved: reserved for kernel driver use
78  * @packet_virt: packet virt address
79  * @size: render algrithm config packet size
80  */
81 struct dsp_render_params {
82         u32 type;
83         u32 reserved;
84         u32 packet_virt;
85         u32 size;
86 };
87
88
89 /*
90  * dsp_user_work - This struct is used by user ioctl
91  *
92  * @magic: work magic should be DSP_RENDER_WORK_MAGIC
93  * @id: work id
94  * @result: work result, if success result is 0
95  * @render: render params
96  */
97 struct dsp_user_work {
98         u32 magic;
99         u32 id;
100        u32 result;
101
102        struct dsp_render_params render;
103 };
```

DSP Kernel Driver Debug Tips

- 跟踪 DSP 驱动 Log：

  echo 0xffffffff > /sys/kernel/debug/dsp/debug

- 查看 DSP 状态：

  io -4 -l 64 0x620bfc00

- 查看 DSP 是否挂起：

  io -4 -l 8 0x33400680

- 查看 Mailbox 消息：

  io -4 -l 128 0x33810000

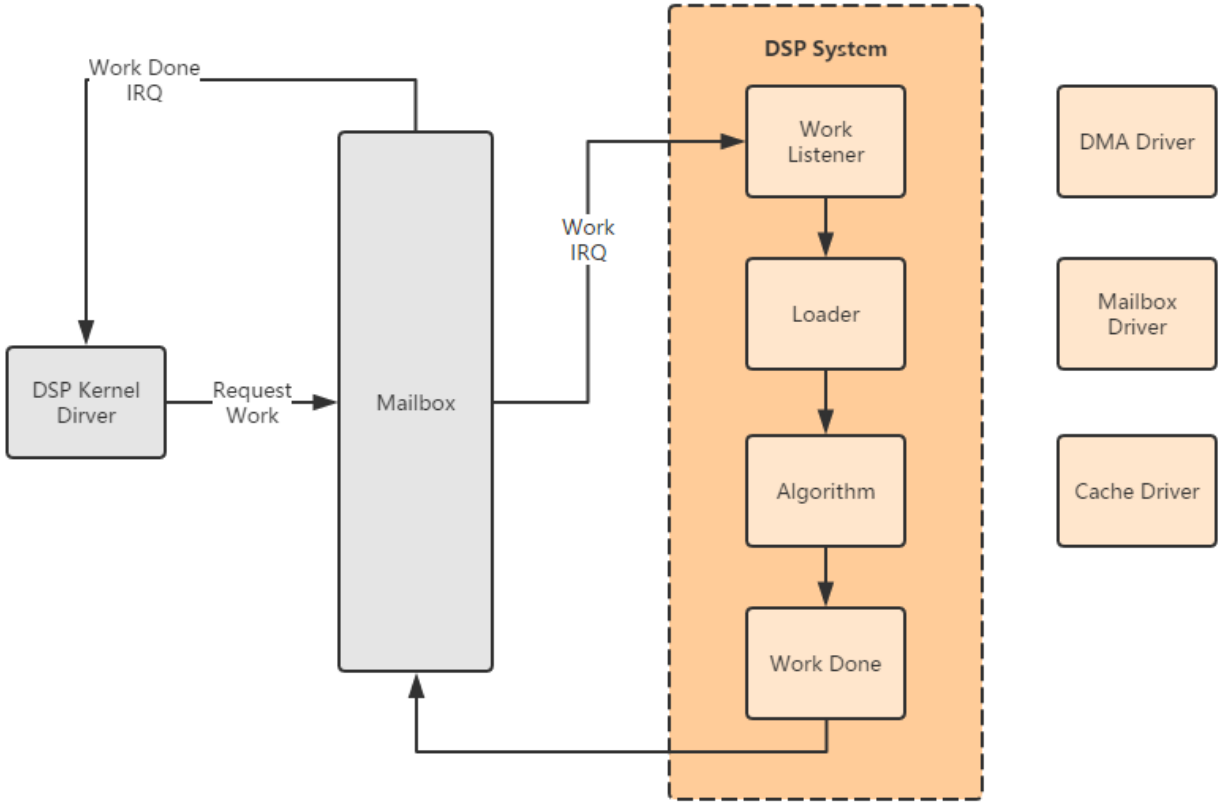
DSP 系统介绍

DSP System Introduction

DSP System 运行在 DSP 内部的 PTCM 上，它的主要功能如下：

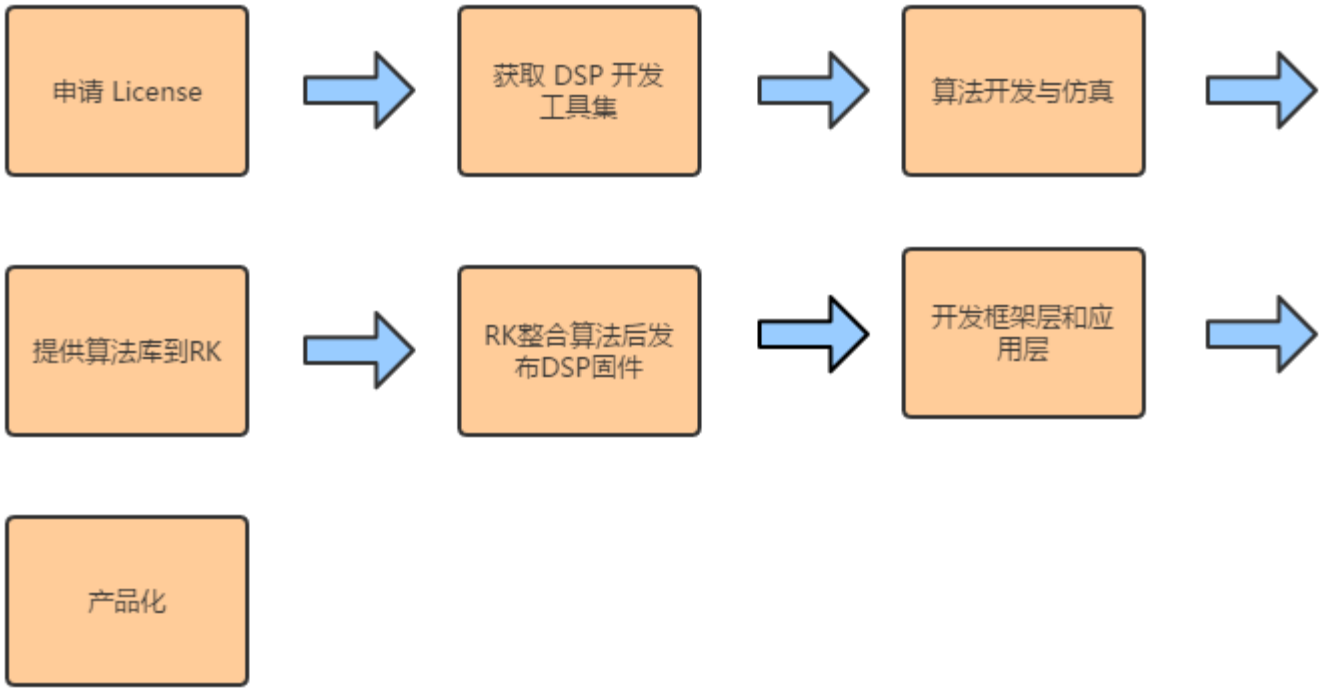
- 监听来自 Mailbox 的工作请求
- 负责加载不同算法的代码和数据
- 调用执行算法
- 通过 Mailbox 将处理结果返回给 ARM

DSP System Architecture



DSP 开发说明

DSP Algorithm Development Workflow



DSP Algorithm Development Tips

1、算法中的所有代码段和数据段都必须按 RK 提供的段名定义好

2、DSP 内部资源有限，仅将占用大部分运算量的代码和数据放到 DSP 内部的 TCM，其他代码和数据均放到外部 DDR，PTCM 的限制为 28K，DTCM的限制为 118K

3、得到 License 后，RK 会提供一个算法开发的基础工程，里面包含必要的库和驱动，建议客户基于我们的基础工程开发算法

4、算法调试完成，可以用 RK 提供的脚本生成算法的 Library 文件

5、Library 的接口统一定义格式为：int do_your_algorithm(void *params);

6、DSP 固件目前由 RK 打包提供，特殊情况下，客户也可以自行通过 RK 提供的固件打包工具生成 DSP 固件

7、开发适配算法的框架层是可选的，客户也可以在应用层直接调用 DSP 驱动来实现 DSP 算法的调用

如何开发 DSP 算法？

由我们的核心算法同事在另外一个课程介绍，大致包含如下内容：

- CEVA-XM4 架构介绍
- 算法的实现
- 算法在 DSP 上的优化

Q&A

Thank you!