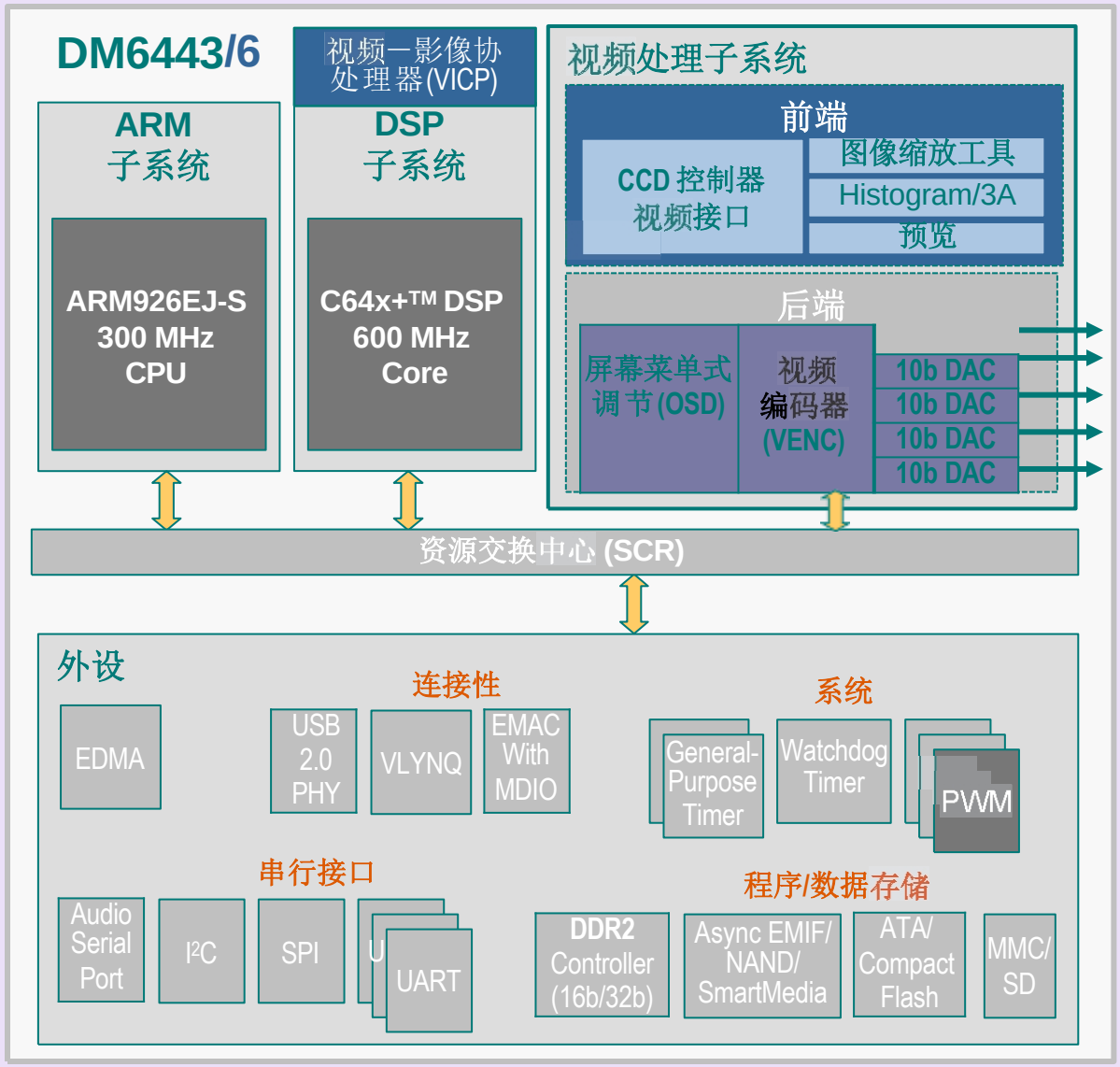
TMS320DM644x™ 处理器 框图



双核的外设分配

由上图可以看出，DM6446提供：

两个内核（DSP和ARM），视频处理子系统（VPSS），两个电源域，多个时钟树，多个引脚独立和复用的外设。

对于双核的达芬奇架构，大家最关心的是两个核之间的资源分配，通信方式。在DM6446中

ARM独享（DSP不可用）的外设有：

UART0/1/2，I2C，看门狗定时器，PWM0/1/2，ARM中断控制器， USB2.0，ATA/CF，SPI，VPSS，GPIO，EMAC/MDIO，EMIFA， VLYNQ，MMC/SD。

DSP独享（ARM不可用）的外设有：

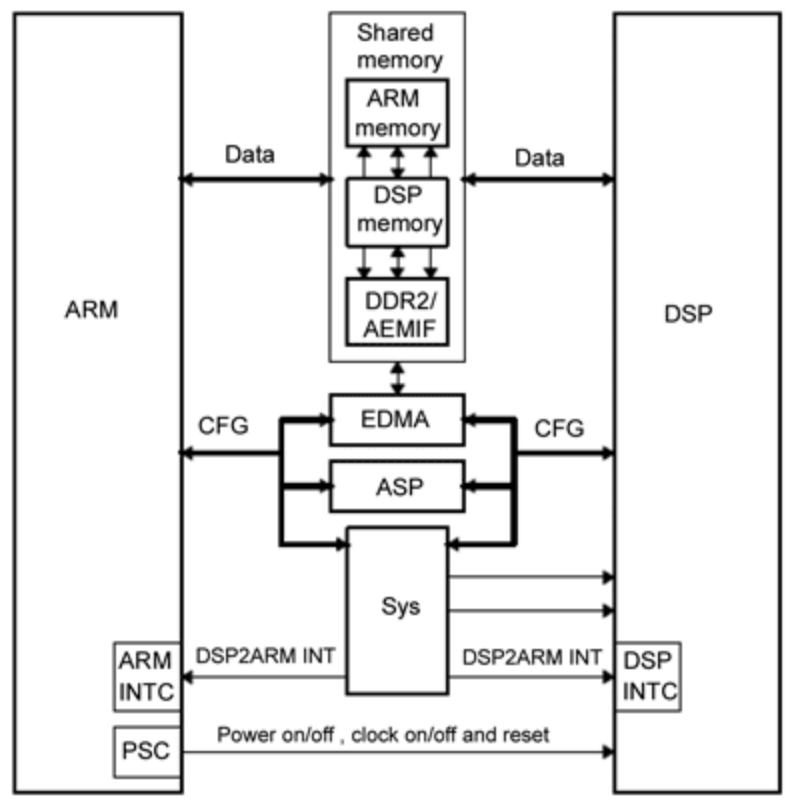
DSP中断控制器，VICP

ARM和DSP共享的外设有：

EDMA，TIME0/1，Power Sleep Controller，ASP，EMIF Data

如下图，可以很清楚的看到ARM可以访问DSP片内存储器（L2RAM和L1D，L1P），DSP可以访问ARM片内存储器，ARM和DSP共享DDR2和AEMIF。

ARM可以中断DSP（4个通用中断和1个不可屏蔽中断），DSP可以中断ARM（2个通用中断）。ARM控制DSP的电源、时钟，复位和引导。



- DM6446的初始化顺序

- 1，DM6446复位

  芯片的配置由PSC决定，取决于BTSEL[0:3]， EM_WIDTH， AEWA和DSP_BT的状态。

- 2，ROM Boot Loader（如果被选）

  NAND和UART0的初始化。

- 3，引导加载

  以U-BOOT为例

  1. 使能电源域：DDR2和DSP；
  2. 设置时钟频率（ARM，DSP和DDR2时钟乘除系数）；
  3. 设置引脚复用控制器；
  4. 设置ARM引导启动操作系统

- 4，操作系统启动

  以LINUX为例

  1. 初始化ARM；
  2. 初始化硬件系统；
  3. 初始化LINUX环境

软件架构：ARM + DSP

- ARM为主处理器：用户应用程序在ARM实现

  移植操作系统OS：LINUX、WinCE……

  用户用下列3个APIs来构建自己的应用程序：

  EPSI：Easy Peripheral Software Interface设备驱动程序　VISA：Video, Imaging, Speech , Audio应用层音／视频编解码引擎接口
  xDM：xDAISfor Digital Media具体的音／视频编解码算法接口，由VISA调用

- DSP为从处理器：主要用来实现视频／图像处理

  ARM与DSP之间用DSP/BIOS LINK来通信

  DSP主要用来实现视频／图像编解码算法xDM

操作系统和驱动程序

- OS：已移植了基于MontaVista内核的LINUX 2.6.10
- EPSI：基于LINUX的设备驱动程序

  串口：UART、IIC、SPI
  存储：ATA、NAND、MMC
  网络：10/100M以太网
  USB：海量存储，主机端和设备端驱动程序
  音频：OSS音频驱动程序
  视频：V4L2视频采集，FBDev/DirectFB视频显示

Codec Engine

- 达芬奇技术体系中引入了Codec Engine，并创建了一整套的应用开发平台。Codec Engine为通用处理器(GPP)上的开发者提供更为简单的开发环境。
- Codec Engine是一系列用于表示和运行数字多媒体标准化DSP算法接口(XDAIS)及算法的API。XDAIS定义了一整套的多媒体算法编程接口，可单独在GPP或DSP上运行，也可在DSP上运行，而GPP通过Codec Engine对其实行控制。对于所有支持的运算器结构、运行方式及操作系统，Codec Engine都有相同的API。Codec Engine定义了4类编解码器算法接口标准，分别是视频、图像、语音、音频，简称VISA。

VISA

- 复杂的信号处理层通过音／视频编解码引擎和VISA API进行抽象
- VISA API是音／视频编解码的用户接口
- VISA用来处理视频Video、图像Image、语音Speech、音频Audio
- 编码与解码的API组相互独立，所以总有8组API：

  VIDENC、IMGENC、SPHENC、AUDENC
  VIDDEC、IMGDEC、SPHDEC、AUDDEC

- 每个VISA组中关键的API有：

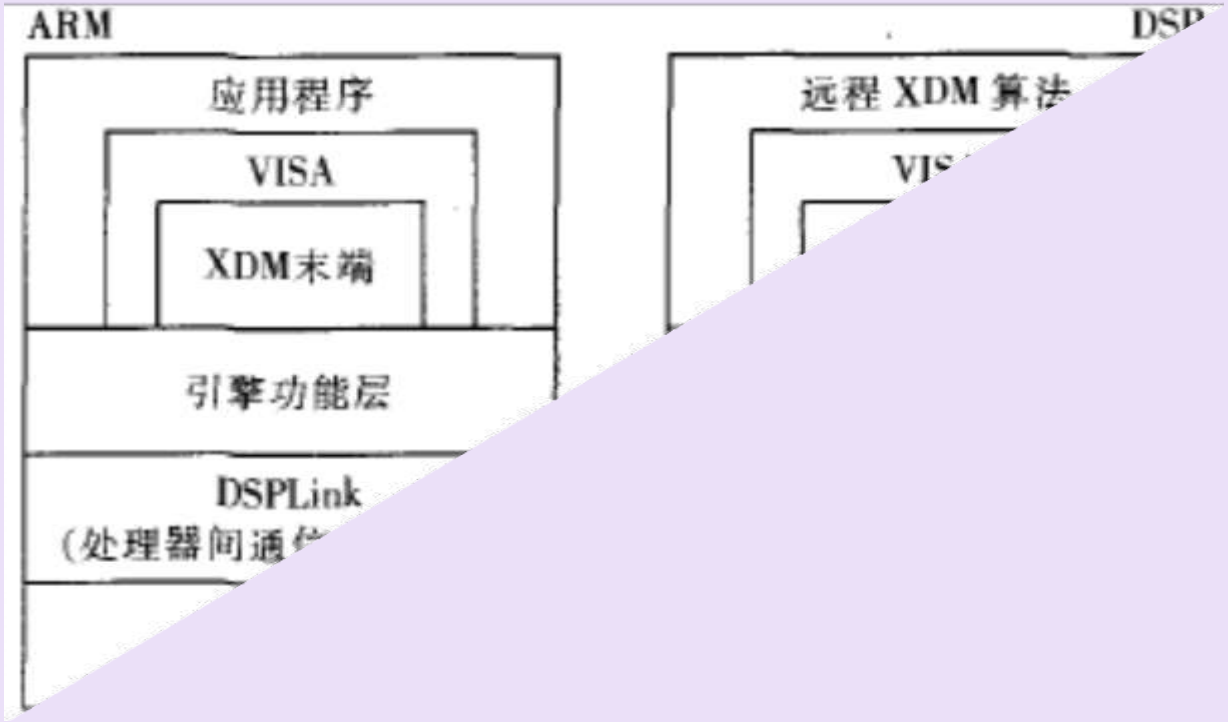
  xxx_create
  xxx_process
  xxx_control
  xxx_delete

Codec Engine框架分析

- Codec Engine在解决达芬奇双处理器架构问题时首先引入了远程过程控制(RPC)的概念。RPC是指在另一个处理器上远程执行一个命令或过程。RPC有客户端和服务端，客户端通过某种通信协议并最终通过物理链路向服务端发送一个命令，服务端执行命令并通过相应的通信方式将结果返回给客户端。

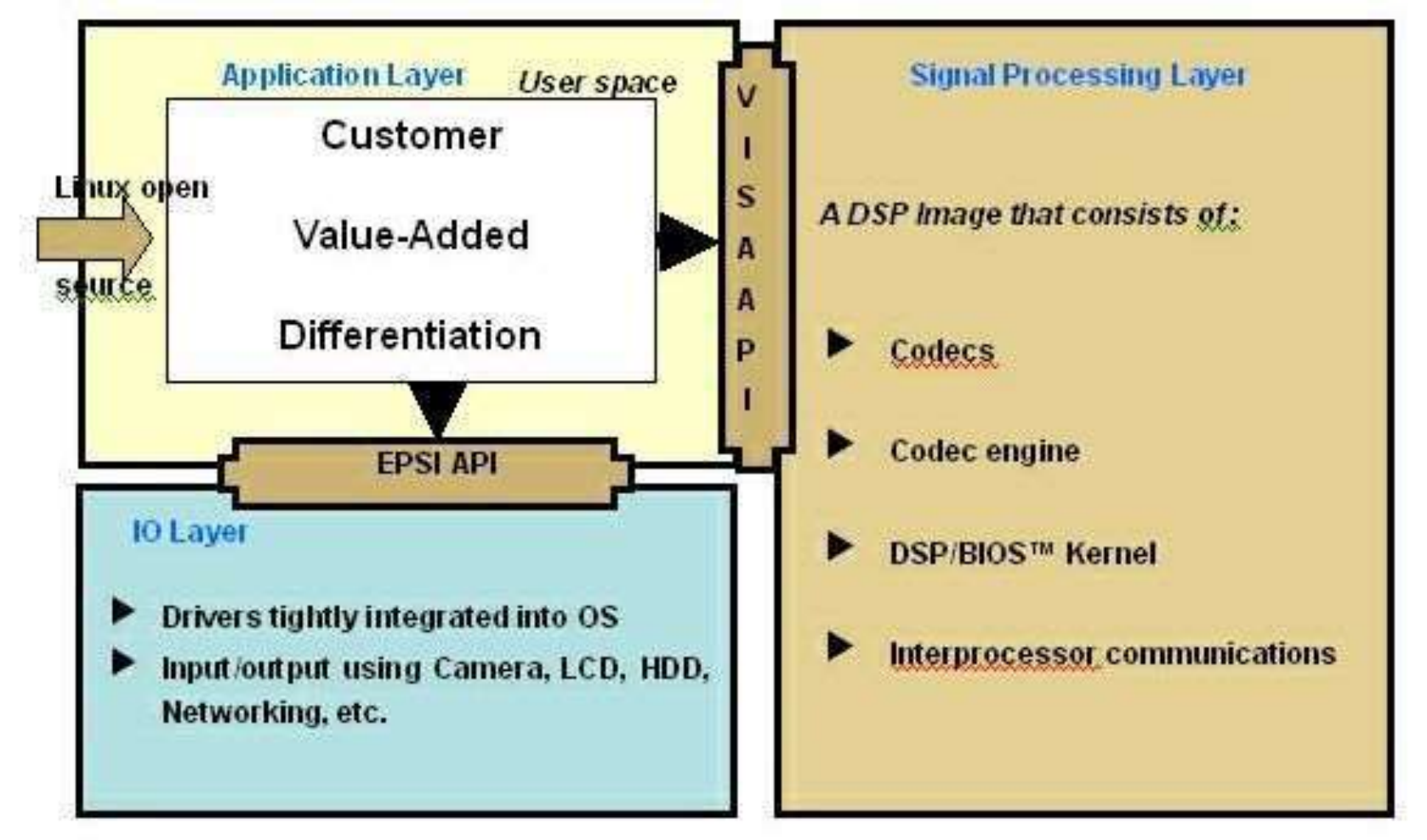
- 达芬奇DM6446芯片将ARM作为客户端、DSP作为服务端，两者之间的物理链路就是两个处理器之间共享的DDR2存储器，而将物理链路上的通信协议称为DSPLink。

- 下图给出了这种框架的结构示意图。图中，共享DDR2存储器是ARM与DSP之间共享的内存，通过它进行物理数据的交换。DSPLink是达芬奇架构双处理器间通信的基础软件，为开发人员提供了一个通用的API，用于描述ARM 与DSP之间通信的物理链路的特征。引擎功能层用于管理算法对象的实例，例如创建一个对象的具体过程等。VISA层是面向引擎功能层的一个接口，用于定义创建、删除和使用一个特定符合XDM(XDAIS的扩展，专用于多媒体应用)标准的算法的进程。由于Codec Engine的主要任务是为从ARM发起的VISA命令控制远程的XDM算法起到一个管道的作用，因此VISA层实际上就是代表了XDM接口层。

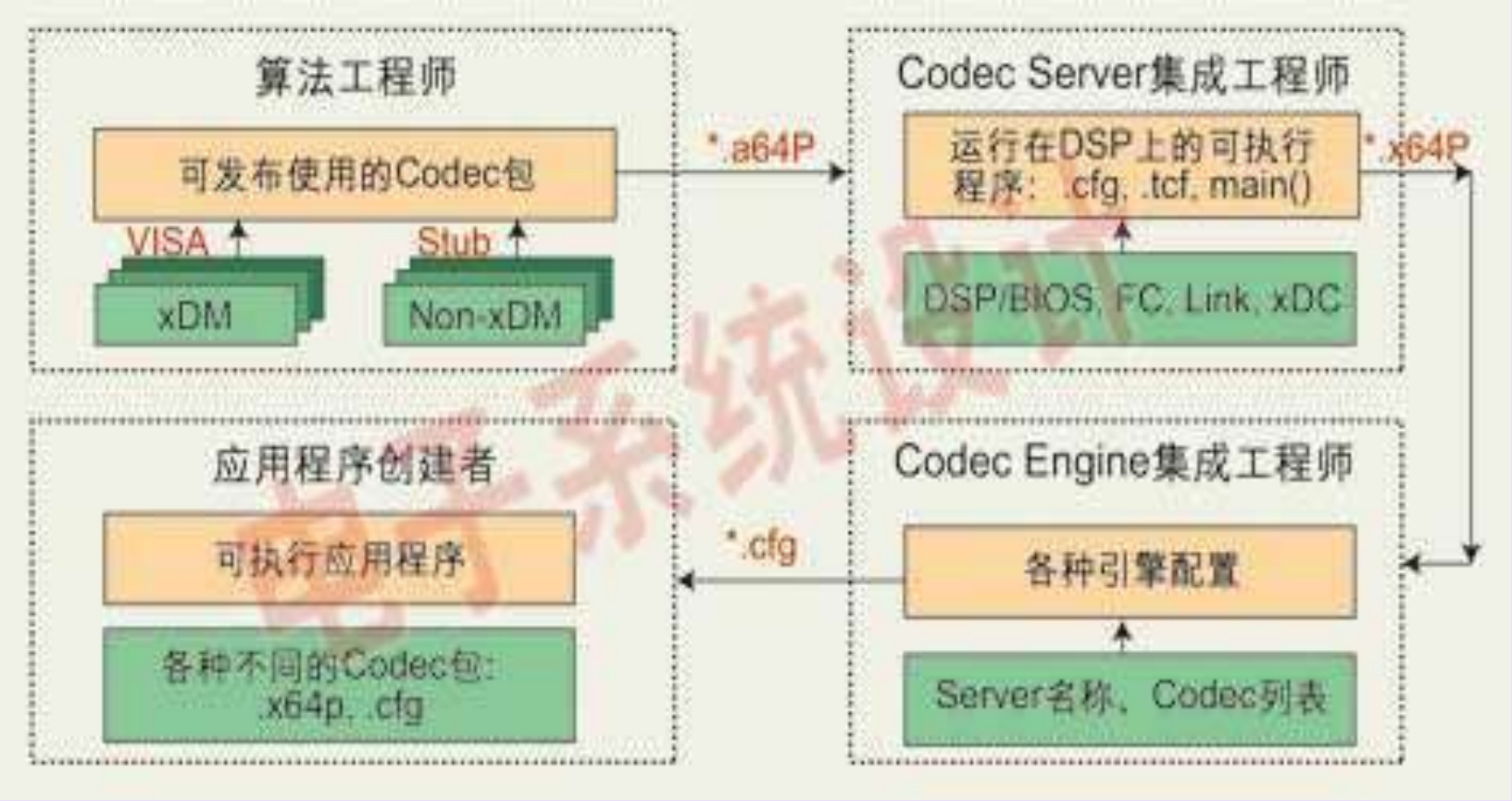


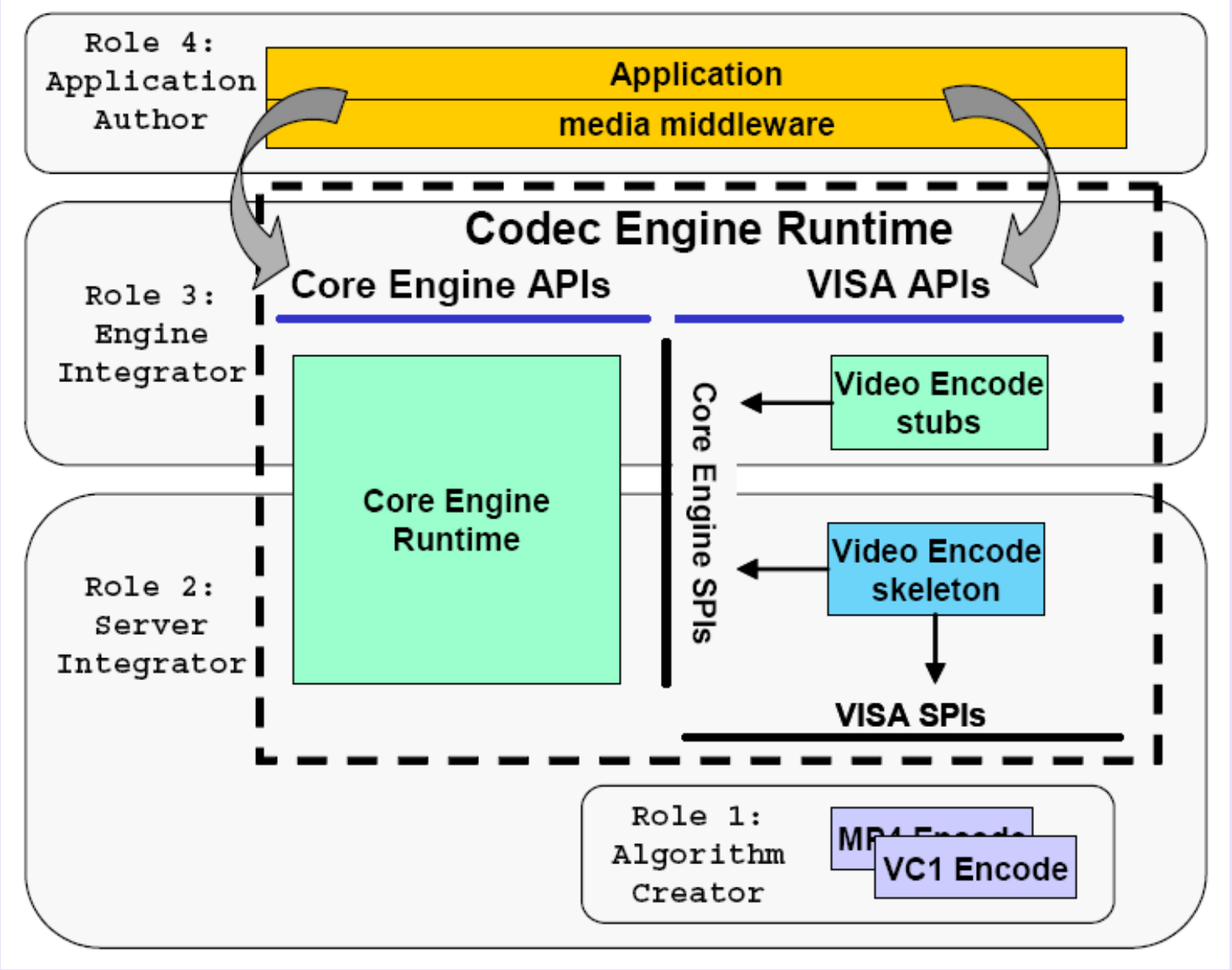
- 一般来讲，软件系统分为应用层、信号处理层和I/O层三部分，TI提供的达芬奇参考软件框架就是基于这样的结构，如图所示。Davinci的应用工程师可以在系统的用户空间在系统功能性上添加和发挥自己的特色。信号处理层通常都运行在DSP一侧负责信号处理，包括音视频编解码算法、Codec Engine、DSP的实时操作系统DSP/BIOS及和ARM通信的模块。I/O层就是我们通常所说的驱动，是针对Davinci外设模块的驱动程序。

- 其中应用层通过Codec Engine的VISA(Video, Image, Speech, Audio)API来调用DSP侧的算法，通过EPSI(Easy Peripheral Software Interface)API来访问和操作Davinci的外设。


Application Layer
User space
Customer
Value-Added
Differentiation
Linux open
source
EPSI API
VISA API
Signal Processing Layer
A DSP Image that consists of:
Codecs
Codec engine
DSP/BIOS™ Kernel
Interprocessor communications
IO Layer
Drivers tightly integrated into OS
Input/output using Camera, LCD, HDD, Networking, etc.

- 如下图所示，DaVinci的软件开发通常需要四个步骤：




Role 4:
Application
Author
Application
media middleware
Codec Engine Runtime
Role 3:
Engine
Integrator
Core Engine APIs
VISA APIs
Video Encode
stubs
Core Engine SPIs
Core Engine
Runtime
Role 2:
Server
Integrator
Video Encode
skeleton
VISA SPIs
Role 1:
Algorithm
Creator
VC1 Encode

- 前面我们提到，应用工程师通过调用Codec Engine的API来调用和运行符合xDAIS的算法。在Davinci软件中，符合xDAIS的音视频编解码算法(即xDM算法)的调用是通过Codec Engine的VISA API完成的。Codec Engine通过这套API为算法的执行提供了一个标准的软件架构和接口。

- Codec Engine是介于应用程序和具体算法之间的软件模块，其中的VISA API通过stub和skeleton访问Engine SPI最终调用具体的算法。因此，Codec Engine的工作是通过完成VISA API的任务来体现的。VISA API分为四部分VISA create/control/process/delete，我们以codec算法运行在DSP为例，通过VISA API的执行过程了解Codec Engine的工作原理。

- 在调用VISA API之前需要在应用程序中通过Engine_open()这个Engine API把DSP的可执行程序加载到DSP的memory，同时把DSP从复位状态释放，这时DSP开始运行DSP Server的初始化程序在DSP侧创建一个优先级最低的任务RMS(Remote Management Server)，RMS负责管理和维护对应到具体codec算法的Instances。应用程序调用VISA create API，相应的VISA create函数到Engine SPI中的Codec table中查到这个codec运行在远端DSP侧。

- 接着Engine SPI通过OSAL(Operating System ion Layer)、DSP Link把VISA create的命令传到DSP侧的RMS。RMS通过DSP侧Engine SPI的codec table找到要调用的codec算法后，就会在RMS中创建一个相应的Instance(即一个DSP/BIOS系统中的任务)。VISA create会返回一个Instance的Handle，以便于给这个Instance做后续的VISA control/process/delete提供信息。VISA delete和VISA create原理类似，只是RMS删除掉相应的codec算法的Instance和执行codec算法的任务。

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：https://d.book118.com/575131110244011040