Reserved	[15:13]		0	X	X
Line_Yoffset_ Pr	[12:0]	当浏览行改变时，目标图像屏幕上跳跃像素的数值。当非交错 YCbCr4:2:0/4:2:2 格式时，可以使用浏览行 CB 补偿区。	0	X	0

21 多格式视频编码器

FIMV-MFC 1.0 版本是一个高能的视频编解码器 IP，它支持 H.263P3， MPEG-4 SP，H.264 和 VC-1。FIMV-MFC 1.0 版本由嵌入式位处理器和视频编解码器核心模块组成。该位处理器解析或构成位流，并控制视频编解码器。加快位流处理，在位处理器中包括一些硬件加速器。通过 AMBA APB 总线和 AMBA AXI 总线下载位处理器的程序和数据。如图 21-1 所示，显示 FIMV-MFC 1.0 版结构框图。

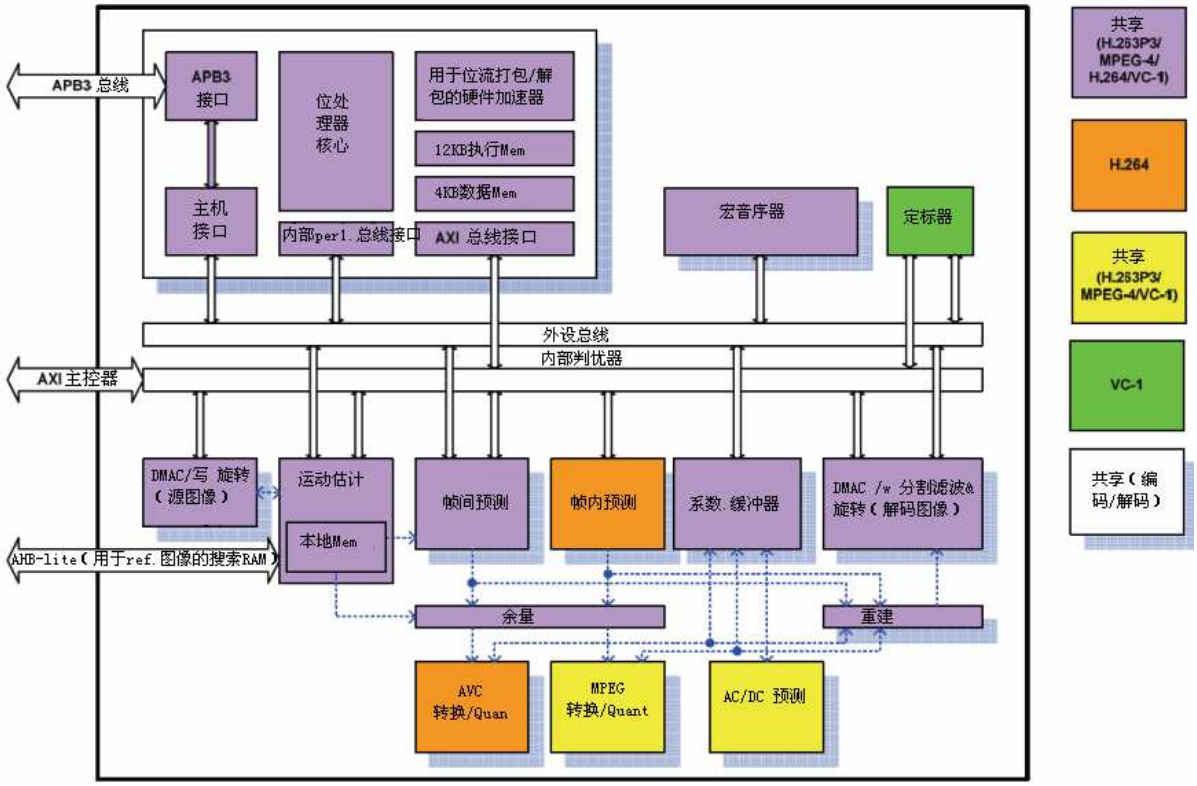

图 21-1 FIMV-MFC 1.0 版结构框图

FIMV-MFC V1.0 视频编解码器被优化，以减少逻辑门的数目。它具有共用大部分的子模块，用于多重标准。运动估值模块采用搜索 RAM 来减少外部 SDRAM 上的带宽。一般来说，运动估值通过多次读取相关像素数据。运动估值模块加载的相关像素数据来自外部 SDRAM，并将它们存储到搜索 RAM 中。通过 AMBA AHB 存取搜索 RAM。FIMV-MFC 1.0 版本包括一个旋转/镜像模块。在编码器旋转和/或镜像初始图像时，无需请求附加的带宽。然而在解码器中，带有任一旋转和/或镜像的解码图像都将被写入外部存储

器。

内部 AXI 判优器模块作出公断请求，从内部 DMA 控制器到用户 SoC。

如图 21-2 所示， 描述位处理的任务，视频编解码器的核心模块以及如何连接应用软件。基本上，主机处理器以帧的形式通过提供的 API 和 FIMV-MFC 1.0 进行通信。给视频编解码器更大的灵活性和调试的能力，与位流相关的所有的处理被指派到位处理器中。

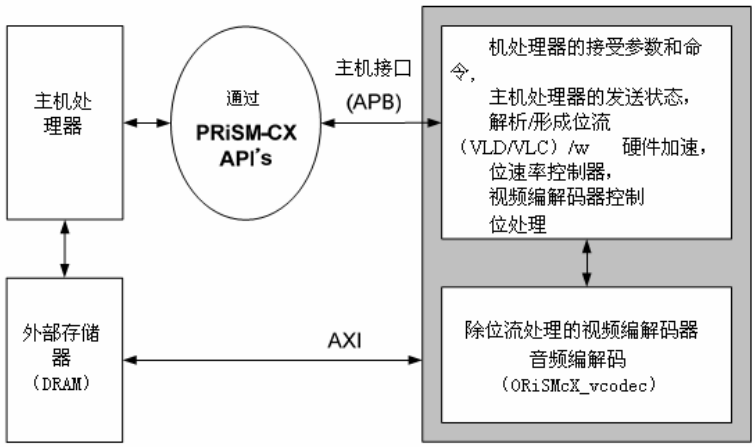

图 21-2 位处理器的任务和视频编码解码器模块

21.1 特征

FIMV-MFC 1.0 版本是一个多标准视频编解码器的 IP，它能够处理个别编解码器元件的 H.263P3，MPEG-4 SP，H.264 BP 和 VC-1 MP。

FIMV-MFC 1.0 版本包括以下功能：

1．多标准视频编解码器

（1）MPEG-4 部分 2 应用简单文件的编码/解码。

（2）H.264/AVC 基线应用文件编码/解码。

（3）H.263 P3 编码/解码。

（4）VC-1（WMV9）主要应用文件解码。

（5）支持多方呼叫。

如：同时发生 1 流解码和 3 流解码是可能的。

（6）支持多格式。

如：视频 IP 同时编码成 MPEG-4 位流和解码成 H.264 位流。

2．编码工具

（1）[-16,+16] 1/2 和 1/4 像素精度的运动估值。

（2）支持所有可变块的大小。

在编码情况下，不支持 8×4，4×8 和 4×4 块大小。

（3）无限制的运动向量。

（4）MPEG-4 交流/直流预测。

（5）H.264/AVC 的帧内预测。

（6）支持 H.263 的附件 I，J，K(RS=0 和 ASO=0)和 T。

在编码情况下，不支持附件 I 和 K（RS=1 或 ASO=1）。

（7）错误修复工具

MPEG-4 重新同步。关于 RVLC 的标记和数据分割。

宏之间的位/宏的固定数量。

H.264/AVC FMO 和 ASO

（8）位速率控制（CBR&VBR）。

（9）CIR(循环内部刷新)/AIR(适应内部刷新)。

3．前/后旋转/镜像

（1）编码器中，8 旋转/镜像模式用于导入图像。

（2）解码器中，8 旋转/镜像模式用于输出图像。

4．可编程

（1）FIMV-MFC 1.0 版本嵌入 16 位 DSP 处理器，它是专注于处理位流和控制编解码器的元件。

（2）用于主机处理器和视频 IP 通信的通用寄存器和中断。

5．性能

（1）向上全双工 VGA 30fps 编码/解码。

（2）向上半双工 720x480 30fps（720×576 25fps）编码/解码。

6．易集成性

（1）用于主机处理器通信的 AMBA，32 位 APB(w/ PREADY)接口。

（2）用于外部存储器 AMBA，32 位 AXI 接口。

21.2 位处理器

本节描述位处理器，这是在各种不同格式下优化过程的位流，如 MPEG-4，H.263，H.264 和 VC-1。

该位处理器是一个嵌入式的可编程 16 位 DSP，它可以高度优化处理位流数据。除处理位流之外，位处理器通过主机接口控制视频编码解码器和主机处理器的通信。该处理器有 12KB 的程序存储器和 4KB 的数据存储器。

如图 21-3 所示，显示位处理器的结构框图。专用的寄存器包括指令，中断和代码下载寄存器。通用寄存器，64 个 32 位寄存器，可以用于主机处理器发送参数给视频编码解码器。如果应用程序需要超过 64 个寄存器，一个外部存储器可以用于扩展，因为位处理器能通过 AXI 总线接口存取外部存储器。

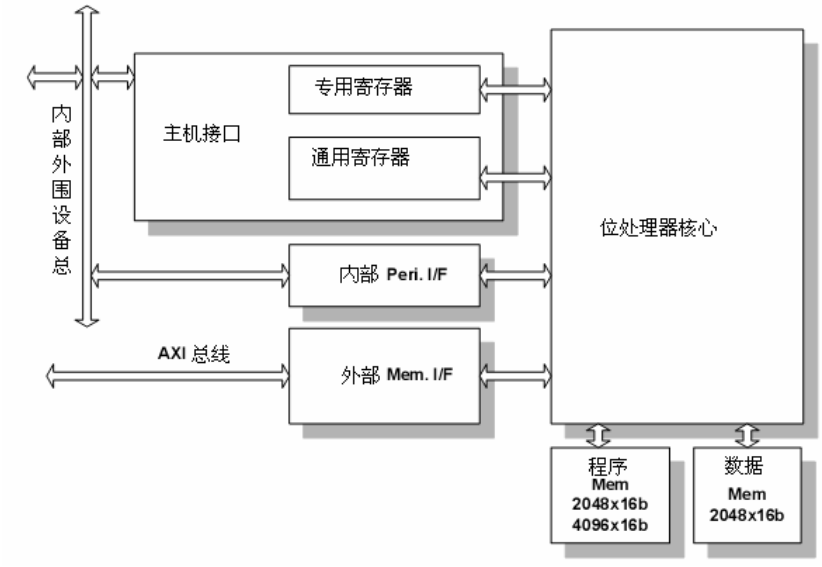

图 21-3 位处理器的结构框图

硬件加速

该位处理器核心嵌入硬件加速子模块如下。

- 加速器支持位流打包的指令，如 put_bits。
- 加速器支持位流未打包的指令，如 get_bits 和 show_bits。

- 用于 VLC 和 VLD 操作的查表和搜索模块。
- 运动向量预测/重建模块。
- DMA 控制器从/到外部的 SDRAM 传输位流。

下载固件

驱动视频编解码模块的位固件和主机处理器接口被分成两部分。一部分用于通过 APB 总线，主机处理器下载的启动代码。启动代码大小是 1024 字节。另一部分用于编解码，如 MPEG-4，H.263，H.264 和 VC-1。用于下载固件的执行程序，在初始化步骤中只执行一次。

1. 启动代码

运行编解码器之前，主机处理器必须下载 BIT 启动代码到程序存储器中，2048×16 同步单端口 SRAM ，通过主机接口如下。

第 1 步: addr = 0 ；

第 2 步: code_data = （地址< < 16 ） + bit_code [地址]；

第 3 步: 将 code_data（数据代码）写入到主机接口的 CodeDownLoad 寄存器，该接口嵌入到位处理器中；

第 4 步: 地址=地址+ 1；

第 5 步: 如果（地址< 512 ）到第 1 步，否则转到程序下载编解码器固件。

bit_code 是一个数组，其大小是 512×16 位。

注：上述 bit_code[0-511]必须被写入主机接口的 CodeBufAddr 寄存器中的指定区域。

2．编解码器固件

除启动代码之外，操纵 IP 需要一个固件的封装。基本上，封装是用于提供 MPEG-4，H.263P3，H.264 和 VC-1 的编解码器。必须写固件到外部存储器的区域并发送关于区域的低级的地址，通过写入到 CodeBufAddr 寄存器。在运行时序时，固件部分自动加载到内部位存储器中。如图 21-4 所示，显示位固化存储空间框图。

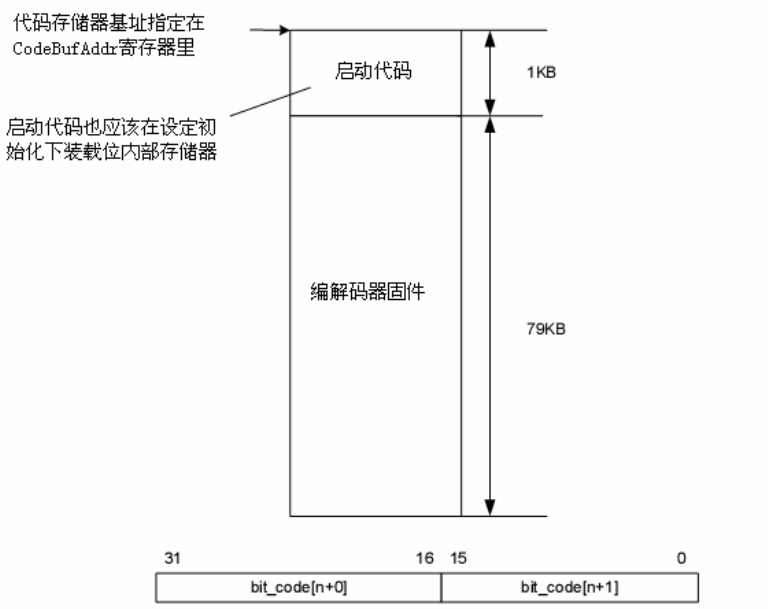

图 21-4 位固件的存储空间

以下程序是在视频编解码器中下载编解码器固件初始化的步骤：

第 1 步: addr = 512 ；

sdram_addr = *CodeBufAddr + 1024;（sdram 即动态随机存取存储器）

第 2 步: code_data = (bit_code[地址] << 16) + bit_code[地址+1];（code_data 代码数据）(bit_code 位代码）

第 3 步: *sdram_addr = code_data;

第 4 步: 地址=地址+ 2 ；

第 5 步: sdram_addr = sdram_addr + 4。

第 6 步: 如果（地址< （ 80 × 1024 ） ）回到第 1 步，否则完成下载。

3．运行编解码器

该部分描述位处理器如何控制视频编码解码器和通信主机处理器。

所提供的固件可以同时处理 8 个进程。每个进程可以有不同的格式，MPEG-4，H.362P3，H.264 或 VC-1 和编解码器进程中的编码或解码。举例来说，可能同时处理一个 MPEG-4 的编码进程，一个 H.264 编码进程中，一个 H.263 P3 解码进程和一个 VC-1 的解码进程。

编码图像和/或解码位流进程如下：

- 创建进程：可以创建和配置进程。

- 运行进程：在一个适当的时间场合，可以执行特定的进程。适当的时间场合意思是，当编解码器是空闲状态和在外部存储器中的图像编码或位流解码准备就绪。
- 结束进程：可以退出某个具体的进程。

如图 21-5 所示， 举例说明运行编解码器的固化状态。

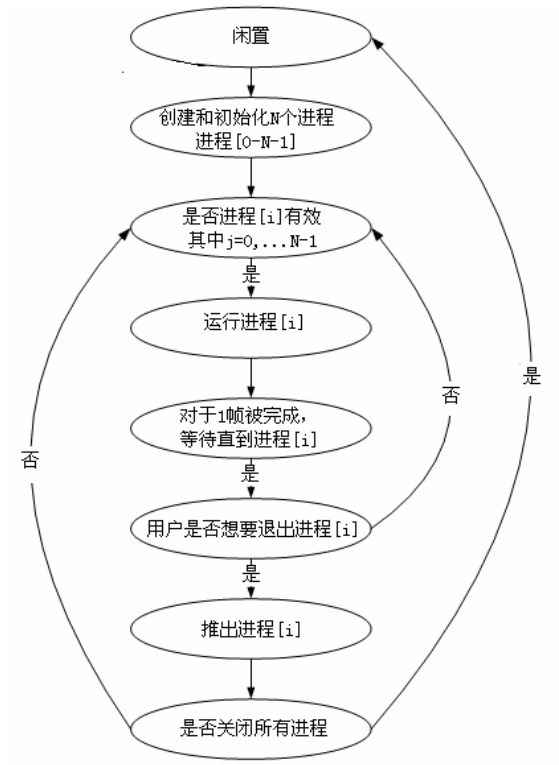

图 21-5 编解码器固件状态图

4．进程 ID-运行索引

每个进程的建立，以特定的 ID 命名，运行索引范围为 0～到 7。

基本上，ID 被分配基于创建的顺序。在创建过程初始化步骤后，主机处理器命令位处理器执行关于运行索引的指定进程。

如果进程创建顺序为（a）MPEG-4 编码，（b）H.264 编码，（c）适用于位流 A 的 H263P3 解码，（d）适用于位流 B 的 VC-1 解码，分配 ID 如下。

MPEG-4 编码：RunIndex = 0。

H.264 编码：RunIndex = 1。

适用于位流 A 的 H.263P3 解码：RunIndex = 2。

适用于位流 B 的 VC-1 解码：RunIndex = 3。

5．分配编码格式 - RunCodStd

除进程 ID 外，RunCodStd 是用来在创建过程中定义编码标准和是否创建进程编码图像或解码位流。

RunCodStd = 0: MPEG-4/H.263P3 编码。

RunCodStd = 1: MPEG-4/H.263P3 编码。

RunCodStd = 2: H.264 解码 。

RunCodStd = 3: H.264 编码。

RunCodStd = 4: VC-1 解码。

例如：如果进程创建一个例子，RunIndex 和 RunCodStd 的每个进程列举如下：

(a) MPEG-4 编码进程： RunIndex = 0, RunCodStd = 0。

(b) H.264 编码进程： RunIndex = 1, RunCodStd = 3 。

(c) H.263P3 解码进程： RunIndex = 2, RunCodStd = 1 。

(d) VC-1 解码进程：RunIndex = 3, RunCodStd = 4。

以下为多格式视频编解码器中 Mpeg4，H264，H263 和 Vc1 的初始化代码。请大家参考：

```
void MFC_InitProcessForDecodingMpeg4(
    u32 uProcessIdx, u32 uStreamBufStAddr, u32 uStreamBufSize,
    u32 uFrameBufStAddr, bool bDecRotEn, bool bMp4DeblkEn)
{
    MFC_InitProcessForDecoding(uProcessIdx, MP4_DEC, uStreamBufStAddr, uStreamBufSize,
        uFrameBufStAddr, bDecRotEn, bMp4DeblkEn, false);
}
void MFC_InitProcessForDecodingH264(
    u32 uProcessIdx, u32 uStreamBufStAddr, u32 uStreamBufSize,
    u32 uFrameBufStAddr, bool bDecRotEn, bool bH264ReorderEn)
{
    MFC_InitProcessForDecoding(uProcessIdx, AVC_DEC, uStreamBufStAddr, uStreamBufSize,
        uFrameBufStAddr, bDecRotEn, false, bH264ReorderEn);
```

```
}
void MFC_InitProcessForDecodingH263(
    u32 uProcessIdx, u32 uStreamBufStAddr, u32 uStreamBufSize,
    u32 uFrameBufStAddr, bool bDecRotEn, bool bH263ReorderEn)
{
    MFC_InitProcessForDecoding(uProcessIdx, AVC_DEC, uStreamBufStAddr, uStreamBufSize,
        uFrameBufStAddr, bDecRotEn, false, bH263ReorderEn);
}

void MFC_InitProcessForDecodingVc1(
    u32 uProcessIdx, u32 uStreamBufStAddr, u32 uStreamBufSize, u32 uFrameBufStAddr, bool
bDecRotEn)
{
    MFC_InitProcessForDecoding(uProcessIdx, VC1_DEC, uStreamBufStAddr, uStreamBufSize,
        uFrameBufStAddr, bDecRotEn, false, false);
}
```

6．编解码器状态

FIMV-MFC V1.0 提供繁忙标志寄存器和中断，该中断可以给出是否以指定的格式编码或解码一帧完成的信息。当 IP（低于）运作，BusyFlag 读取‘1’。 如果编码或解码一帧被完成，同时 IP 是在等待状态，可以从主机处理器接受一个命令，BusyFlag 变为‘0’。 IP 发生中断，在这个时间，BusyFlag 变为‘1’。请求下一个进程前，主机处理器必须清除中断。

如果在处理一帧过程中有一些错误，BusyFlag 和中断也有效。主机处理器可以知道，是否一个过程被正常地终止或不被正常地终止，是通过提交主机接口的其它状态寄存器决定。

7．中断

1）到一个主机处理器

位处理器，可以产生中断请求到主机处理器。基本上，这个中断是用来显示编码或解码一个帧的完成。中断信号 IREQ 高位有效，直到主机处理器通过将主机接口的中断清除寄存器写“1”来清除它。