支持 MBAFF 的 H.264/AVC 解码器运动矢量预测模块设计

包 磊,周开伦,林 涛

(同济大学 超大规模集成电路研究所,上海 200331)

摘 要:分析了宏块自适应帧场模式(MBAFF)在 P 帧和 B 帧的帧间预测算法,提出了可行的数据组织结构和硬件实现方法。作为完整的解码器的一部分,其 RTL 代码已经完成了功能验证和仿真,证明该设计是行之有效的。

关键词:H.264;MBAFF;帧间预测;运动矢量预测

中图分类号:TP335+.2 **文献标识码**:A

Motion vector prediction algorithm and hardware scheme of H.264/AVC MBAFF decoding

BAO Lei, ZHOU Kai Lun, LIN Tao

(Institute of Very Large Scale Integrated Circuit, Tongji University, Shanghai 200331, China)

Abstract: This paper analyzed the motion vector prediction algorithm of H.264/AVC MBAFF decoding and proposed the data structure and hardware scheme. As a part of the whole decoder, the RTL code of this module has been verified and proved to be a feasible design.

Key words: H.264; MBAFF; frame inter estimation; motion vector prediction

H.264/AVC 支持 3 种图像编码模式:帧模式、场模式和宏块自适应帧场(MBAFF)[1]模式。在帧模式下,1 幅图像被划分成由 16×16 宏块组成的帧;在场模式下,1 帧图像的顶场和底场被划分成由 16×16 宏块组成;在 MBAFF 模式下,帧场编码的选择在宏块级指定,1 帧图像被划分成由 32×16 的宏块对组成,每 2 个宏块组成的宏块对以帧模式或者场模式编码。一段图像可能既有动态的区域又有静态的区域,H.264 的 MBAFF 模式可以根据图像的每个区域选择最佳的编码模式。一般来说,MBAFF 模式对于视频编码的效率比其他两种模式更好,但编码和解码的复杂度也更高。本文将探讨在解码的过程中 MBAFF 模式在 P 帧和 B 帧的帧间预测算法和硬件设计。

1 帧间预测技术

在进行帧间预测时,空间上距离较近的图像区域往往具有相关性很强的运动矢量,而且相对于以前视频压缩标准中最小 8×8 的划分,H.264 最小 4×4 的划分使得较小的图像区域的运动矢量具有更高的相关性,因此,可以利用预测技术预测出 1 组运动矢量 MVp,而编码器只需要传送能量很小的实际值和预测值的差值,也就是运动矢量残差 MVd,这样就可以提高编码效率。在解码时,解码器只需要采用同样的算法先预测出运动矢量的预测值 MVp,而后将其与残差 MVd 相加便可以恢复出运动矢量的实际值 MV[1]供运动补偿使用。MVp[1]取决于运动补偿的尺寸和邻近 MV 的有无。

在帧间预测模式下,宏块有 16×16、16×8、8×16、8×8、8×4、4×8、4×4 这 7 种运动矢量的分割方法。为方便描述,把参考宏块相应分割的运动矢量命名为 MVLXN[2],参考索引命名为 refIdxLXN[2],其中 N 可以为 A、B、C、D,分别表示当前宏块或者宏块分割的左边、上边、右上和左上的相应宏块分割。当前宏块分割的运动矢量 MVpLX[2]就是通过 MVLXN 和 refIdxLXN 预测得到的。图 1 所示为非 MBAFF 模式时当前宏块分割为 16×16 的情况,E 为当前宏块或宏块分割,A、B、C 分别为 E 的左、

上、右上方的3个相对应分割块。如果E的左边不止1个分割，则取其中最上的1个为A；上方不止1个分割时，取最左边1个为B。

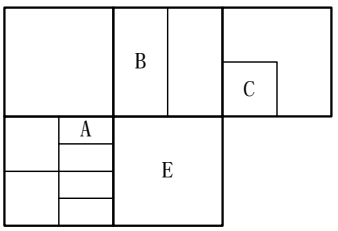

图1 宏块相邻分割示意图

1.1 预测运动矢量MVp的计算

在宏块分割为16×16、8×8、8×4、4×8和4×4时，运动矢量的预测值是由参考宏块分割A、B、C的运动矢量计算得到的：

(1)如果分割B和C不可用，而分割A可用，则会对MVLXB、MVLXC、refIdxLXB、refIdxLXC重新赋值：MVLXB=‘MVLXA、MVLXC=MVLXA、refIdxLXB=refIdxLXA、refIdxLXC=refIdxLXA。

(2)根据refIdxLXA、refIdxLXB和refIdxLXC的取值计算mvpLX：

①如果分割A、B、C的参考索引refIdxLXA、refIdxLXB或refIdxLXC中的一个等于当前分割的索引号refIdxLX，则当前分割的矢量预测值由相应分割的运动矢量得到：MVpLX=mvLXN。

②如果分割A、B、C的参考索引refIdxLXA、refIdxLXB或refIdxLXC都不等于当前分割的索引号refIdxLX，则通过取A、B、C的运动矢量中值得到：

MVpLX[0]=Median(MVLXA[0],MVLXB[0],MVLXC[0])

MVpLX[1]=Median(MVLXA[1],MVLXB[1],MVLXC[1])

运动矢量的预测在当前宏块分割为16×8和8×16的时候会先做如下的判断，如果不满足则通过上述取中值的方法得到预测值：

(1)对于1个宏块被分割成2个16×8子宏块的情况

①计算mbPartIdx等于0的子宏块如图2(a)所示。如果refIdxLXB等于当前分割的refIdxLX，则当前块的矢量预测值由B宏块的相应分割的矢量得到：MVpLX=MVLXB。

②计算mbPartIdx等于1的子宏块如图2所示，如果refIdxLXA等于当前分割的refIdxLX，则当前块的矢量预测值由A宏块的相应分割的矢量得到：MVpLX=MVLXA。

(2)对于1个宏块被分割成2个8×16子宏块的情况：

①计算mbPartIdx等于0的子宏块如图2(b)所示。如果refIdxLXA等于当前分割的refIdxLX，则当前块的矢量预测值由B宏块的相应分割的矢量得到：MVpLX=MVLXA。

②计算mbPartIdx等于1的子宏块如图2(b)所示，如果refIdxLXC等于当前分割的refIdxLX，则当前块的矢量预测值由C宏块的相应分割的矢量得到：MVpLX=MVLXC。

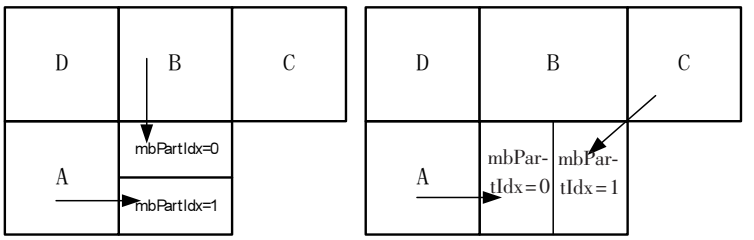

图2 分割运动矢量预测

1.2 参考宏块分割的运动矢量MVLXN和参考索引refIdxLXN的计算

在计算参考宏块分割A、B、C的运动矢量MVLXN和参考索引refIdxLXN之前，应先检测分割C是否可用，如果C不可用，则用分割D的信息代替。A、B、C的运动矢量MVLXN和参考索引refIdxLXN的计算：

(1)如果宏块mbAddrN不可用或者宏块mbAddrN为帧内编码，或者mbAddrN相应分割或者子宏块分割的predFlagLX等于0，则对MVLXN置0，对refIdxLXN置-1。

(2)MVLXN为mbAddrN相应宏块分割或者子宏块分割的运动矢量，refIdxLXN为mbAddrN相应宏块分割的参考索引值：

MVLXN=MvLX[mbPartIdxN][subMbPartIdxN]

refIdxLXN=RefIdxLX[mbPartIdxN]

(3)在MBAFF模式时，需要对MVLXN和refIdxLXN再做1次计算：

①如果当前宏块为场宏块，而mbAddrN为帧宏块，则：

MVLXN[1]=MVLXN[1]/2

refIdxLXN=refIdxLXN×2

②如果当前宏块为帧宏块，而mbAddrN为场宏块，则：

MVLXN[1]=MVLXN[1]×2

refIdxLXN=refIdxLXN/2

1.3 空间相邻参考宏块分割的选择[2]

在非MBAFF模式时，当前宏块的空间相邻宏块地址A、B、C、D的位置如图3所示。空间参考宏块mbAddrN可以是MbAddrA、MbAddrB、MbAddrC、MbAddrD或者CurrMbAddr。

在MBAFF模式时，当前宏块的空间相邻宏块地址A、B、C、D的位置如图4所示。此时会根据当前宏块是帧宏块或者场宏块currMbFrameFlag以及当前宏块是顶

MbAddrD	MbAddrB	MbAddrC
MbAddrA	CurrMbAddr	

图3 空间相邻宏块地址

MbAddrD	MbAddrB	MbAddrC
MbAddrA	CurrMbAddr 或	
	CurrMbAddr	

图4 MBAFF空间相邻宏块地址

宏块或者底宏块 mbIsTopMbFlag 来计算 1 个中间值 mbAddrX，并根据 mbAddrX 的帧场信息 mbAddrXFrameFlag 来得到最终的参考宏块 MbAddrN。参考宏块 mbAddrN 可以是 MbAddrA、MbAddrA+1、MbAddrB、MbAddrB+1、MbAddrC、MbAddrC+1、MbAddrD、MbAddrD+1、CurrMbAddr 或者 CurrMbAddr-1。

参考分割 A、B、C、D 取自参考宏块 mbAddrN，也可能来自当前宏块的空间相邻宏块或可能是当前宏块内已解码的分割。

1.4 运动矢量预测所需数据的组织

运动矢量的预测需要当前分割的左边、上边、左上和右上相应分割的 MVLXN 和 refIdxLXN 信息来计算得到当前分割的 MVLX。如果宏块的 4 个子宏块都采用 4×4 分割，则 1 个宏块有 16 个运动矢量，即每个 4×4 block 有 1 个运动矢量。这样需要储存每个 block 的运动矢量和参考索引。1 个宏块需要存储 16 个 block 的运动矢量 MVLX 和参考索引 refIdxLX，当采用其他分割类型时，每个分割内所有 block 的运动矢量和参考索引为相同值。

帧间预测按照 block 存储和使用数据。当前宏块分割的运动矢量和参考索引需要按照 block 来存储以用于后面宏块的预测。每个分割内所有 block 的运动矢量和参考索引为相同值，所以参考分割的 MVLXN 和 refIdxLXN 的引用也可以按照 block 来进行。

因为同一分割内所有 block 的运动矢量和参考索引为相同值，因此，参考分割的 MVLXN 和 refIdxLXN 也就是参考宏块相应的 block 的 MVLX 和 refIdxLX。图 5 为非 MBAFF 模式时预测当前分割的运动矢量的示意图。由图可见，在进行预测时是通过与当前分割最接近的左、上、左上以及右上的 4 个 4×4 block 的运动矢量来对当前的运动矢量进行预测的。图 5(a)~(d)分别表明了当前宏块采用不同划分时

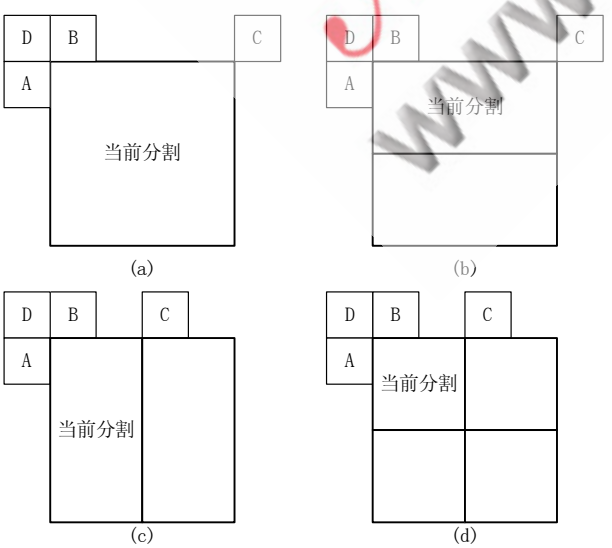

图 5 参考块位置选择

A、B、C、D 4 个参考块的取值情况。在预测时一般选取 A、B、C 块对当前的运动矢量进行预测，当 C 不可用时则选用 A、B、D 进行预测。在 MBAFF 模式时，由于参考宏块的位置不同(参见 1.3 节)会作相应变化。

2 直接预测模式和硬件设计

H.264/AVC 对运动矢量的重建引入了直接预测模式，在该模式下运动矢量残差没有被传送，只有宏块的预测模式会被传送。解码器根据预测模式以及其余宏块的信息对当前的运动矢量进行恢复。运动矢量的重建流程如图 6 所示。在直接预测模式下，H.264 分别采用 3 种不同的方式对运动矢量进行重建。

(1)复制模式。在 P 类型的宏块中，利用空间上相邻分割的运动矢量对当前的运动矢量进行预测。

(2)空间模式。在 B 类型的宏块中，利用空间上相邻分割的运动矢量对当前的运动矢量进行预测。

(3)时间模式。在 B 类型的宏块中，通过将时间上相邻而且在空间上位置相同的 8×8 分割的运动矢量进行重量化来预测出当前的运动矢量。

其中复制模式和空间模式在算法上具有一定的相似性，差别在于复制模式只需要重建 1 个方向的运动矢量，而空间模式需要重建双向的运动矢量。

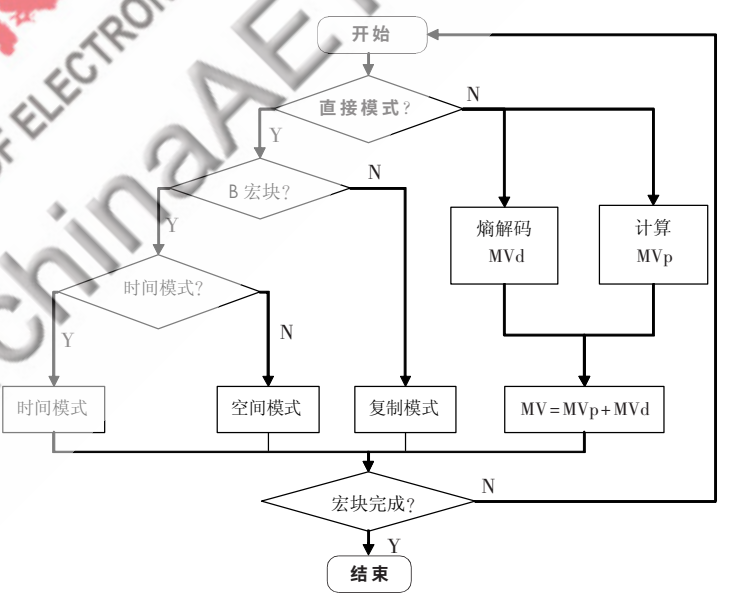

图 6 运动矢量重建流程图

2.1 空间模式及复制模式重建算法和硬件设计

在这 2 种模式下：(1)宏块只能划分成 1 个 16×16 分割或者 4 个 8×8 分割。在划分成 4 个 8×8 时，每 1 个 8×8 分割可以独立地被设置成直接模式或者 ReadMV 模式；(2)计算当前的运动矢量需要借助在空间上相邻分割的运动矢量来对当前分割的运动矢量进行重建。此时运动矢量的预测算法与前面介绍的预测运动矢量的算法基本相同，所不同的是此时宏块中的任意 1 个分割预测时所使用的相邻块信息均由图 7 所示的 A、B、C、D 4

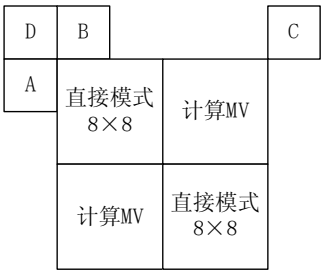

图 7 矢量重建参考块位置图

个块预测。

Direct 8×8 用 A、B、C、D 4 个块进行预测。预测的基本算法流程如图 8 所示。由图可见，在预测时首先检测参考块 C 是否可用，如不可用，则 C 块的参数用 D 块的参数代替；而后利用 A、B、C 块的参考索引 RefIdxA[f/b]、RefIdxB[f/b]、RefIdxC[f/b]中大于 0 的最小值来预测出当前分割的参考索引号 ReFrame[f/b]，如果 RefIdxA[f/b]、RefIdxB[f/b]、RefIdxC[f/b]全部小于 0，则 ReFrame[f/b]取 -1；然后用 ReFrame[f/b]与 A、B、C 块的参考索引进行对比，如果 ReFrame[f/b]只是 A、B、C 其中的 1 个，现令其为 SelectedBlk 的索引号相同，则当前分割的运动矢量中，(MVx，MVy)取 SelectedBlk 的(MVxS，MVyS)；否则取 A、B、C 块运动矢量的中值。

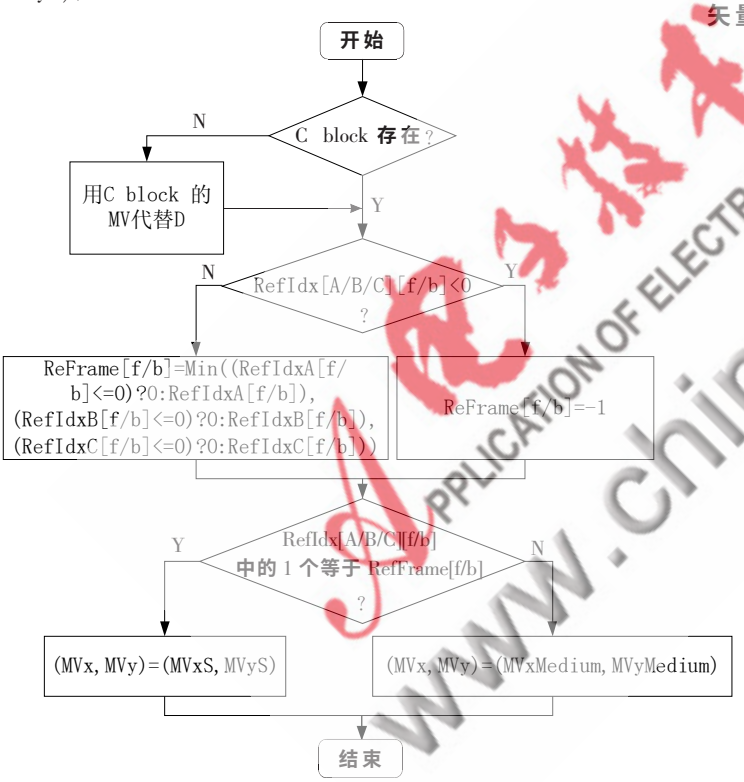

图 8 运动矢量预测算法图

在空间模式及复制模式下，需要使用空间上相邻的上边及左边的 4×4 块的运动矢量信息来对空间模式、复制模式中的运动矢量以及 ReadMV 模式中的 MVp 进行计算。针对非 MBAFF 模式和 MBAFF 模式所需要的 block 的运动矢量信息分析如下：

(1)在非 MBAFF 模式时，硬件实现需要维持 1 个 4×4 block 行，以及当前待解码宏块的左边 4 个 block 列的运动矢量信息。解码器如果需要支持 1 920×1 280 分辨率视频流的解码，则就要存储(1 920/4+4+1)=485 个 4×4 block 的运动矢量信息，如图 9 所示。

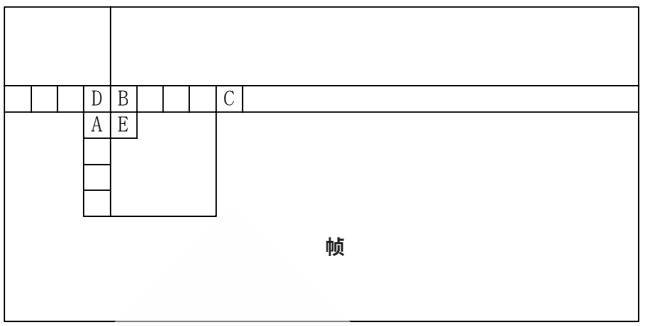

图 9 非 MBAFF 所需块信息

(2)在 MBAFF 模式时，由于宏块对的出现使得所需要的block 信息相比非 MBAFF 模式的情况更加复杂。硬件实现时需要维持 2 个 4×4 block 行，以及当前待解码宏块的左边宏块对的 8 个 block 列的运动矢量信息。解码器如果需要支持 1 920×1 280 分辨率视频流的解码，则就要存储(2×(1920/4)+8+1)=969 个 4×4 block 的运动矢量信息，如图 10 所示。

图 10 MBAFF 所需块信息

设计中将这部分信息存放在片内 SRAM 中。此外，为了便于读取以及节省存取时间，设计中将 1 个 block 的运动矢量信息存放于 SRAM 1 个地址单元中。1 个地址单元的数据结构需要存放前向参考索引 RefIdxF、后向参考索引 RefIdxB、前向水平运动矢量 MVxF、前向垂直运动矢量 MVyF、后向水平运动矢量 MVxB、后向垂直运动矢量 MVyB。

通过以上分析在硬件实现时采用如图 11 所示的结构对该模块进行设计。设计中使用 1 个 Local Sram 作为 1 个Line buffer 来对空间模式及复制模式下需要使用到的空间上相邻块的运动矢量信息进行存储。当该模块被启动后，根据输入的当前宏块的预测模式以及当前宏块的位置信息，地址生成器根据当前宏块的信息产生相应的地址以及控制信号从 Local Sram 中取出 A、B、C、D 4 个 block 的运动矢量信息传送给 MV idx 计算模块来计算当前分割的运动矢量，计算完毕后得出完成信号给地址生成器以便其产生相应的地址将解码好的数据存回 Local Sram 中供后续解码使用。

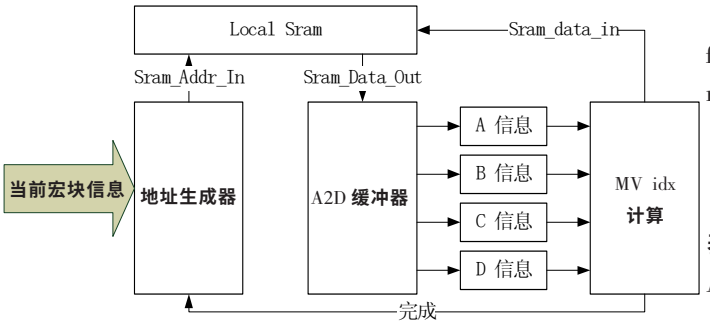

图 11 空间模式硬件结构框图

2.2 时间模式重建算法和硬件设计

图 12 所示为宏块类型为 B 宏块、预测模式为直接预测模式时采用时间模式对运动矢量进行重建的算法示意图。

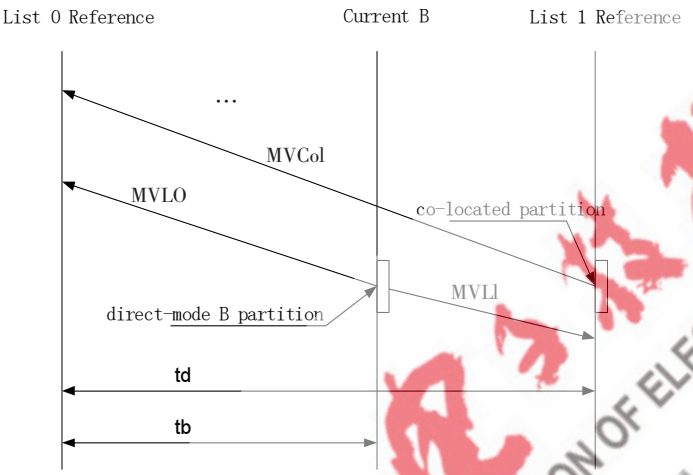

图 12 时间模式矢量重建算法示意图

由图可见,时间模式运动矢量重建的步骤为:

(1)当前待解的分割先在 List[1][0]中找到与自己位置相对应的分割相应的、在 List[0]中的参考图像 List[0][x],将该参考图像作为当前分割的 List[0]参考。

(2)找出 List[1][0]图像中与当前分割位置相对应的分割指向 List[0][x]的运动矢量 MVCol[2]和参考索引 refIdxCol[2]。

(3)根据当前分割所在图像与 List[0][x]图像的播放顺序 POC 的差值 tb[2]、List[0][x]与 List[1][0]的 POC 的差值 td[2],通过如下公式量化出当前分割指向 List[0][x]图像的运动矢量 MVL0 以及指向 List[1][0]的运动矢量 MVL1。

tb = Clip3 (-128, 127, DiffPicOrderCnt (currPicOrField, pic0))

td = Clip3(-128, 127, DiffPicOrderCnt(pic1, pic0))

tx = (16 384 + Abs(td/2))/td

DistScaleFactor = Clip3(-1024, 1023, (tb*tx+32)>>6)

MVL0 = (DistScaleFactor*mvCol+128)>>8

MVL1 = MVL0 - MVCol

(4) 根据 refIdxCol 和当前图像是帧或场图像标志 field_pic_flag 以及当前宏块是帧宏块或场宏块来计算 refIdxL0:

refIdxL0 = ((refIdxCol<0)?0:MapColToList0(refIdxCol))

refIdxL1 = 0

在非 MBAFF 模式时,图像类型可以是 FRM 和 FLD 表示当前是帧或场;在 MBAFF 模式时,图像类型以 AFRM 表示当前图像是宏块级帧场自适应。

在 MBAFF 模式时,首先根据当前图像类型 PicCoding Struct(CurrPic)[2]和参考图像类型 PicCodingStruct(colPic)[2]来计算 1 个宏块地址 mbAddrX,然后再根据 mbAddrX 是否为场宏块以及当前宏块是否为场宏块来计算出在 List[1][0]中与当前分割位置相对应的宏块 mbAddrCol,最后得到其中相应宏块分割 mbAddrCol\mbPartIdxCol\subMbPartIdxCol 的运动矢量 MVCol 和参考索引 refIdxCol,MVCol 和 refIdxCol 取值为 mbAddrCol 相应分割的前向或者后向运动矢量和参考索引。

因此,对于已经解码的图像需要存储每个宏块的运动矢量、参考索引、宏块类型(mbType、subMbType)和宏块的帧场标志 mb_field_decoding_flag[2],以用于当前图像宏块的运动矢量预测。

时间模式下运动矢量重建的硬件实现框图如图 13 所示。设计中采用了空间模式以及复制模式硬件实现所用的 Local Sram。这里还在 Local Sram 中开辟了一段新的存储空间,用于存放所需要的图像层信息。与空间及复制模式一样,重建后的运动矢量信息写回到 Local Sram 中供后续解码使用。

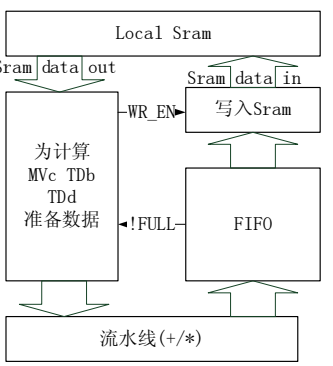

图 13 时间模式硬件结构框图

由运算公式可见时间模式下运动矢量的重建涉及乘法、除法、加法以及减法运算,这些运算对于硬件实现将会带来很大的开销,所以设计中将差值运算拆分成流水线的形式进行运算。

本文介绍了 H.264/AVC 的宏块自适应帧场模式在 P 帧和 B 帧的帧间预测算法,分析了运动矢量预测模块的硬件实现,提出了可行的数据组织结构和硬件实现方法。

参考文献

[1] 毕厚杰.新一代视频压缩编码标准——H.264/AVC[J].北京:人民邮电出版社,2002.

[2] Advanced video coding for generic audiovisual services. ITU-T Recommendation H.264, May. 2005.

(收稿日期:2009-08-12)