

基于最大类间方差法的图像分割系统的设计与实现

王瞰来 杨春玲

(哈尔滨工业大学电气工程及自动化学院, 哈尔滨 150001)

摘要: 在现代战争中, 快速而有效的对攻击目标进行识别和跟踪对获取战争主动权起着很重要的作用, 而要达到这个目的, 就需要从目标图像中准确的分割出目标。在图像分割中, 阈值的选取至关重要。最大类间方差法是一种常用而有效的图像分割算法, 并已在许多实时场合中采用。为满足高速场合的要求, 本文采用 Altera 公司的 Cyclone II 系列的 FPGA 实现类间方差的计算。实验结果表明, 本设计能够实时稳定的对目标分割提取, 分割效果良好。

关键词: 图像分割; 最大类间方差; FPGA

Implementation of image segmentation system based on Otsu method

WANG Jian-lai YANG Chun-ling

(Electrical Engineering, Harbin Institute of Technology, Harbin, 150001)

Abstract: In the modern war, recognizing and tracking the target rapidly and effectively is of vital importance for acquiring war initiative. For the sake, it is necessary to segment the target from the image accurately. In image segmentation, threshold selection is very important. The Otsu's method is an effective algorithm for image segmentation, and has been widely employed in various real-time applications. In this paper, an implementation on FPGA of Altera's Cyclone II series for the BCVC (Between Class Variance Computation) of Otsu's method is presented to meet these high speed requirements. The experimental results indicate that the system can obtain a good performance of image segmentation.

Keywords: Image segmentation; Maximum between class variance; FPGA

引言

图像分割是图像处理和计算机视觉中基本而关键的技术之一, 也是图像理解与模式识别的前提^[1], 在实际中也有着很广泛的应用。例如在医学应用中核磁共振图像的特定器官分割; 在遥感应用中合成孔径雷达图像的目标分割; 在红外技术应用中红外无损检测热图像的分割以及红外成像跟踪系统的目标分割等等。

阈值分割是图像分割中最常用的方法, 通过选取适当的阈值, 将原图像中的目标与背景分开, 为后续的分类、识别提供依据。如何选取最优阈值, 保证

最好的分割效果，一直是阈值分割的难点。人们已经提出了很多阈值选取方法，如 Otsu 提出的最大类间方差法^[2]，Kapur 等提出的最佳熵阈值方法^[1]；1962 年 Doyle 提出的 P-tile 法^[3]是早期的基于灰度直方图的自动阈值选择方法，该方法计算简单，抗噪声性能较好，不足之处是要预先知道给定目标与整幅图像的面积比 P ，因此在 P 未知或 P 随不同图像改变时，该方法不适用。其中 Otsu 在 1979 年提出的通过最大类间方差准则来选取阈值的方法一直被认为是阈值分割的经典算法。本文拟采用 Altera 公司 Cyclone II 系列的 FPGA 实现最大类间方差法以达到图像分割的目的。

1 最大类间方差法（Otsu 法）的基本原理

最大类间方差法（Otsu 法）是由 Otsu 于 1979 年提出的动态阈值分割方法，它的基本思想是利用一幅图像的灰度直方图，依据类间距离极大准则来确定区域分割门限。该方法的基本原理如下：设图像有 L 个灰度级，灰度值是 i 的像素数为 n_i ，则总的像素数是 $N = \sum_{i=0}^{L-1} n_i$ ，各灰度值出现的概率为 $p_i = n_i / N$ ，显然

$$p_i \geq 0 \text{ 且 } \sum_{i=0}^{L-1} p_i = 1。 \text{ 设以灰度 } t \text{ 为门限将图像分割成 2 个区域：灰度级为 } 1 \sim t \text{ 的}$$

像素区域 A（背景类）和灰度级为 $t+1 \sim L-1$ 的像素区域 B（目标类）。A、B 出现的概率分别为

$$p_A = \sum_{i=0}^t p_i, \quad p_B = \sum_{i=t+1}^{L-1} p_i = 1 - p_A$$

A 和 B 两类的灰度均值分别为

$$\omega_A = \sum_{i=0}^t i p_i / p_A, \quad \omega_B = \sum_{i=t+1}^{L-1} i p_i / p_B$$

图像总的灰度均值为

$$\omega_0 = p_A \omega_A + p_B \omega_B = \sum_{i=0}^{L-1} i p_i$$

由此可以得到 A、B 两区域的类间方差

$$\sigma^2 = p_A (\omega_A - \omega_0)^2 + p_B (\omega_B - \omega_0)^2$$

显然， p_A 、 p_B 、 ω_A 、 ω_B 、 ω_0 、 σ^2 都是关于灰度级 t 的函数。

为了得到最优分割阈值，Otsu 把两类的类间方差作为判别准则，认为使得 σ^2 值最大的 t^* 即为所求的最佳阈值：

$$t^* = \text{Arg Max}_{0 \leq t \leq L-1} [p_A (\omega_A - \omega_0)^2 + p_B (\omega_B - \omega_0)^2]$$

因为方差是灰度分布均匀性的一种度量，方差越大，说明构成图像的两部分差别越大，当部分目标错分为背景或是部分背景错分为目标都会导致两部分

差别变小。因此，使类间方差最大意味着错分概率最小，这就是 Otsu 准则。

2 总体方案设计

本系统拟通过 FPGA 对图像传感器采集的图像数据进行存储，针对每一帧图像计算其最大类间方差，从而得到阈值以便分割图像，并把分割后的图像通过 VGA 显示。其中图像分辨率为 320x240，帧频为 45 帧/秒，量化位数为 10 比特。

2.1 原始数据的预处理

为了获得客观景象最详尽的描述，最佳方案是用 3 块 CCD 接收红 (R)、绿 (G) 及蓝 (B) 3 个分量的信息，合成彩色图像。但出于成本与系统体积的考虑，大部分的数字成像设备（如普及型数字相机、数字录像机等）都采用 1 块 CCD 作为接收图像的传感器。一般是在 CCD 表面覆盖颜色滤波阵列 (Color Filter Array, 或称 CFA)，仅允许通过 1 种颜色分量，使 CCD 单元只产生 1 个分量的响应值。

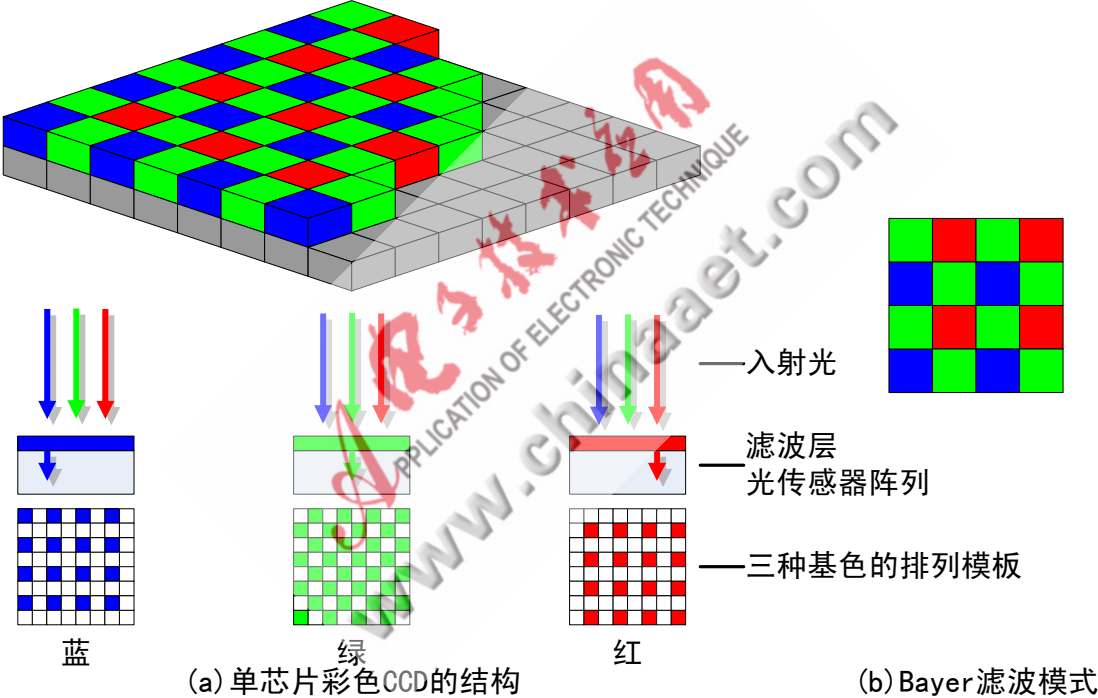


图 1 Bayer 彩色 CCD 结构

Bayer方案由Kodak创立，它把输入光信号分解为RGB三基色，且每个像素点只采集一种颜色光的亮度。如图 1所示，其中绿色像素 (G像素) 最多，占 1/2，红色像素 (R像素) 和蓝色像素 (B像素) 各占 1/4。这是因为人眼对亮度信息最敏感，而绿色像素对亮度的贡献最大。因此，在Bayer滤光镜阵列中，通过绿色滤光镜的光强是通过蓝色滤光镜或红色滤光镜的两倍。这便引出一种通常称作的“4:2:2 RGB”的输出格式，即每发送 2 份红色光和 2 份蓝色光就对应发送 4 份绿色光。这样CCD产生的每个像素都缺少了 2 种颜色，成像设备要获得全彩色图像，就要用周边像素的像素值近似计算出被滤掉的颜色分量，即彩色插值 (Color Interpolation) 或彩色去马赛克 (Color Demosaicing) 处理。

本设计中采用的图像传感器 MT9M011 使用的是 Bayer 类型的颜色滤波阵

列，所以需要设计彩色插值模块，以得到“4: 4: 4 RGB”的输出格式从而进行后续处理。

2.2 亮度信息的获取

通常，我们看到的光不是单一波长的光，而是许多不同波长的光的组合。自然界中的任何一种颜色都可用这三种基本颜色按不同的比例混合得到，它们构成一个三维的 **RGB** 矢量空间。这既是色度学中最基本的原理——三原色原理。任一种颜色和这三种颜色之间的关系可用下面的式子来描述：

任一颜色= R （红色光的百分比）+ G （绿色光的百分比）+ B （蓝色光的百分比）

为了更加有效地压缩图像的数据量以便充分利用传输通道的带宽或者节省存储空间，人们开发了许多颜色空间，**YCbCr** 颜色空间就是其中的一种。由于本设计采用的是彩色图像传感器，获得的图像数据是彩色的，而最大类间方差法所针对的应用场合是灰度图像，所以在把图像数据送入 **Otsu** 模块前，必须要通过颜色空间转换得到相应的亮度信息。

YCbCr坐标与**RGB**坐标之间的关系如式(1)所示，这两个坐标之间的关系如图 2所示。

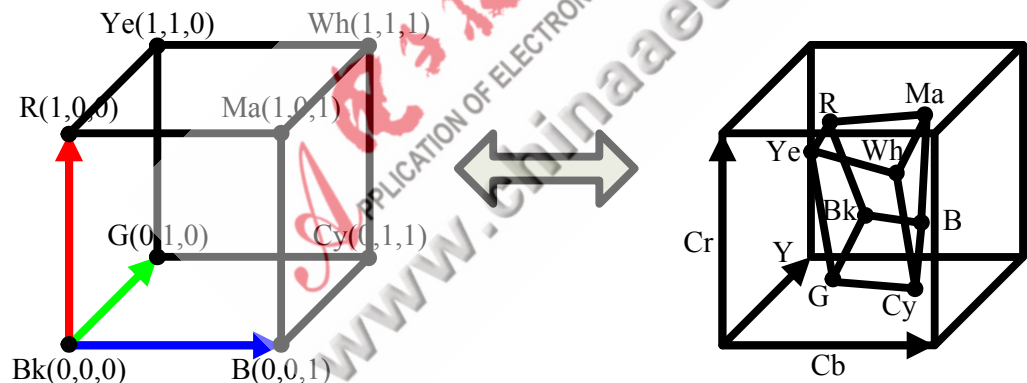
$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0.063 \\ 0.500 \\ 0.500 \end{bmatrix} \quad (1)$$


图 2 **RGB** 坐标与 **YCbCr** 坐标之间的关系

2.3 数据的缓存

由于 **Otsu** 算法是一种基于全局性的算法，通过 **Otsu** 模块计算出的阈值会在当前帧的场消隐期间出现，所以系统需要构建帧存储器以保存当前帧的亮度数据，便于在场消隐期间读出该帧的亮度数据与阈值进行比较并把最终结果传递给 **VGA** 模块以便显示。本系统送入 **VGA** 模块的数字视频图像分辨率为 320x240，其中每个像素值为 10 比特，所以一帧图像的大小为 9.375Kx10 比特，图像尺寸比较大，不宜采用 **FPGA** 片内存储单元构建帧存储器。所以本设计决定采用相对较便宜的高速大容量 **SDRAM** 外围逻辑器件构建帧存储器。

FPGA 设计的指导原则之一就是同步化设计，即对所有时钟控制器件（如触发器、**RAM** 等）都采用同一个时钟来控制。但在实际的应用系统中，随着设计规模的不断扩大，一个系统中往往含有数个时钟，数据不可避免的要在不同

时钟域间传递，完全同步化设计出现困难。在本系统中即有这种情况发生，本系统采用的板级晶振是 50MHz 的，FPGA 收到视频采集命令后，开始接收从 MT9M011 送出的数字视频流。MT9M011 的时钟频率为 25MHz，而 SDRAM 芯片使用的时钟是经过 PLL 将晶振的 50MHz 时钟 2 倍频后的时钟，两者时钟频率不一致。当视频数据从 MT9M011 传入到 SDRAM 时便出现了异步时钟域的问题，本设计采用异步 FIFO 进行数据缓冲。同样的，当在场消隐期间读出该帧的亮度数据与阈值进行比较的时候，由于比较模块采用的时钟频率为 VGA 的时钟频率——25MHz，所以在 SDRAM 与比较模块之间也需要使用异步 FIFO 进行数据缓冲。

由于视频数据的存储和显示是同时进行的，而 SDRAM 存储器是单端口器件，数据的写入和读出不能同时进行，故需要同时进行乒乓操作来完成数据的连续读写。考虑到输入视频流和输出视频流的频率都比较低，所以本设计没有像通常的乒乓操作一样，采用两片相同 SDRAM 芯片，而只采用一片 SDRAM 芯片，通过设计 SDRAM 芯片与系统的接口模块轮流对两块 Bank 进行读写操作，大大节约了资源占用。

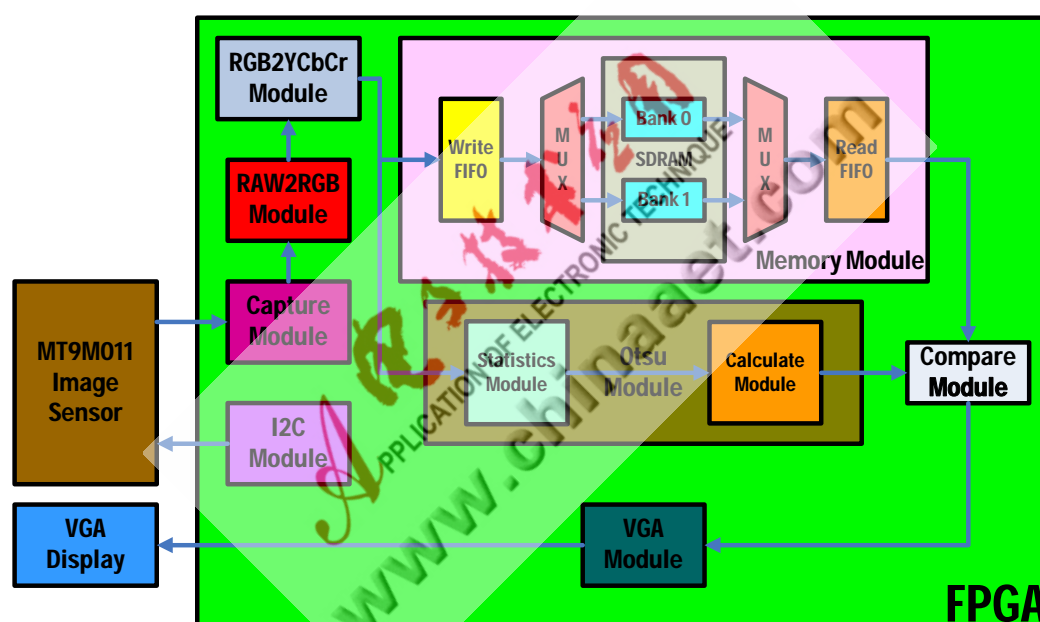


图 3 系统总体设计框图

通过上述分析，本系统的总体设计框图如图 3 所示。FPGA 作为系统的主控芯片，是本系统设计的核心。根据系统的工作需求，FPGA 主控芯片的工作流程为：首先在上电后通过 I2C 模块完成对 MT9M011 图像传感器的初始化配置，选择输出图像的分辨率，帧频及曝光时间等参数。配置完成后，FPGA 等待采集图像的命令。在收到采集命令后，从 CCD 传来的图像信号及其控制信号首先通过 Capture 模块，从而筛选出有效的数据，然后通过 RAW2RGB 模块，利用插值算法得到每个像素点的 R、G、B 数据。因为该系统只针对灰度图像进行分割，所以在送入 Otsu 模块前，需要把 R、G、B 数据送入 RGB2YCbCr 模块，从而得到图像的亮度分量。Otsu 模块包括 Statistics 模块和 Calculate 模块两个子模块。在把该亮度分量送入 Otsu 模块的同时，需把该数据送入 Memory 模块进行存储。Memory 模块用于读写 SDRAM 的 Bank0 和 Bank1，并且可以在两个 Bank 之间进行读写切换操作。在 Otsu 模块输出当前帧的有效阈值后，再读出存储在 Memory 模块中的

亮度数据，在Compare模块中进行比较后，把二值化的结果送至VGA模块供VGA显示器显示。

3.1 图像数据采集模块的设计

该模块实现图像数据的采集，它必须满足图像传感器MT9M011输出数据时序，该时序如表 1及图 4，图 5所示。MT9M011可以通过I2C接口来配置它的帧频，曝光时间，图像分辨率等参数。本设计为提高图像的采样频率以及方便在VGA显示器上显示结果，通过设置寄存器Reg0x03，Reg0x04，使MT9M011的图像分辨率为640x480，其余参数采用默认值。由此时的参数计算出来的帧频为45帧/秒。

表 1 MT9M011 输出数据时序

Parameter	Name	Default Timing at 25MHz	Set Timing at 25MHz
A	Active Data Time	1280 pixel clocks	640 pixel clocks
P	Frame Start/End Blanking	6 pixel clocks	6 pixel clocks
Q	Horizontal Blanking	396 pixel clocks	396 pixel clocks
A+Q	Row Time	1676 pixel clocks	1036 pixel clocks
V	Vertical Blanking	84184 pixel clocks	52184 pixel clocks
Nrowsx(A+Q)	Frame Valid Time	1715840 pixel clocks	496896 pixel clocks
F	Total Frame Time	1800024 pixel clocks	549080 pixel clocks

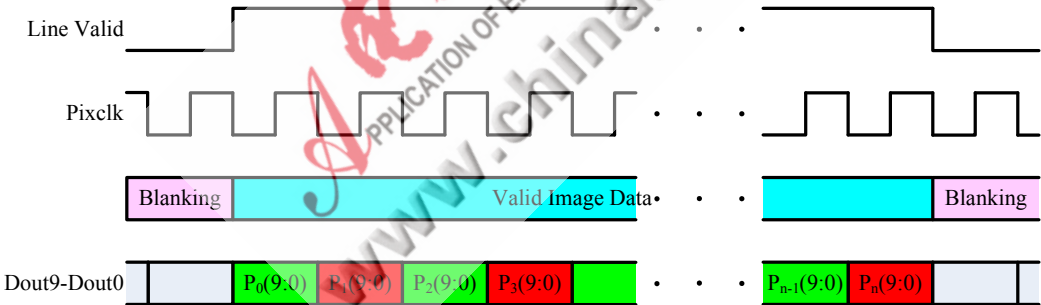


图 4 MT9M011 的像素时序

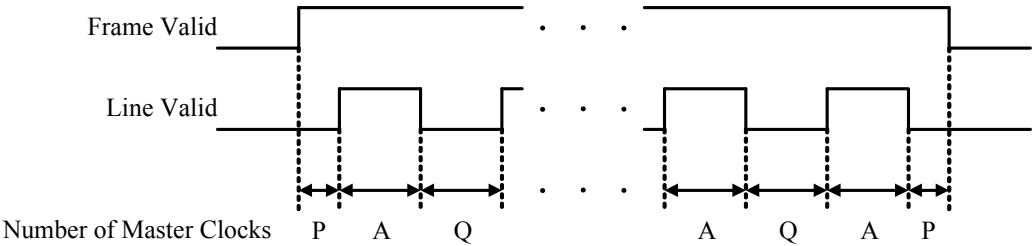


图 5 MT9M011 的行时序及 Frame Valid/Line Valid 信号

当视频捕捉开始键按下时，该模块需要判断这是否是一帧图像的起始，如果是，则开始接收图像数据，否则需要等到下一帧图像开始时才接收数据。只有采取这种措施才能捕捉到完整的视频帧。判断是否是一帧图像的起始，可以通过检测 Frame Valid 信号的上升沿来判断。

需要注意的是，视频捕捉模块在获得有效像素数据的同时也接收了消隐期的图像数据，所以需要设置输出数据有效这个信号，以便在接下来的 RAW2RGB 模块中把有效数据和非有效数据区分开来。

3.2 RAW2RGB 模块的设计

MT9M011 采用的是Bayer型CFA，由于该图像传感器的分辨率为 640x480，为了得到最终显示时的分辨率——320x240，这里采用的插值算法把每四个像素合并为一个像素，像素值的变化如图 6所示，这样经过RAW2RGB模块后，图像的分辨率变为原来的一半，即 320x240。

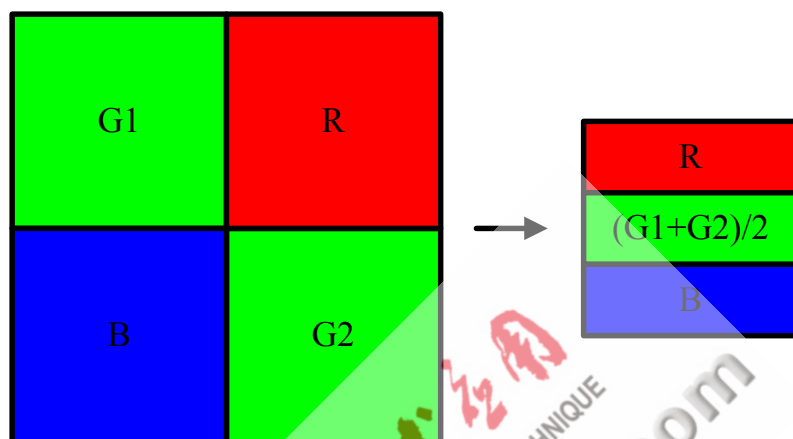


图 6 颜色插值算法示意图

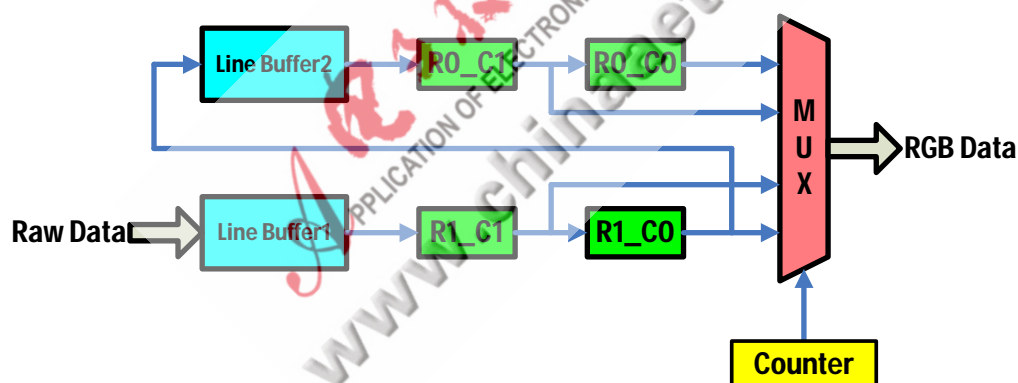


图 7 RAW2RGB 模块的硬件结构框图

该模块的硬件实现框图如图 7所示。当输入数据有效时，Raw格式的数据依次通过Line Buffer1，R1_C1，R1_C0，Line Buffer2，R0_C1，R0_C0 缓存送入MUX单元，得到最终的RGB数据。其中Line Buffer1 和Line Buffer2 的深度为 638，R1_C1，R1_C0，R0_C1，R0_C0 为 4 个D触发器。MUX可以通过计数器来控制，该计数器来自视频捕捉模块的行列像素计数的最低位。当行列计数的最低位都为 0 时，寄存器R0_C0，R0_C1，R1_C0，R1_C1 中的数据应该分别是 G1，R，B，G2 分量，则此时MUX单元应输出的RGB数据可以通过这 4 个寄存器中的数据得到，分别为R0_C1， $(R0_C0+R1_C1)/2$ ，R1_C0。当行列计数的最低位为其它组合时，输出数据无效。通过这种方式，不但得到了每个像素的RGB数据，还使得图像分辨率变为原来的 1/4。

为了检验算法的正确性，特意设定了 8 种典型颜色，并按照颜色插值算法由这 8 种颜色的Raw格式数据预先计算出了相应的RGB数据，如图 8所示。

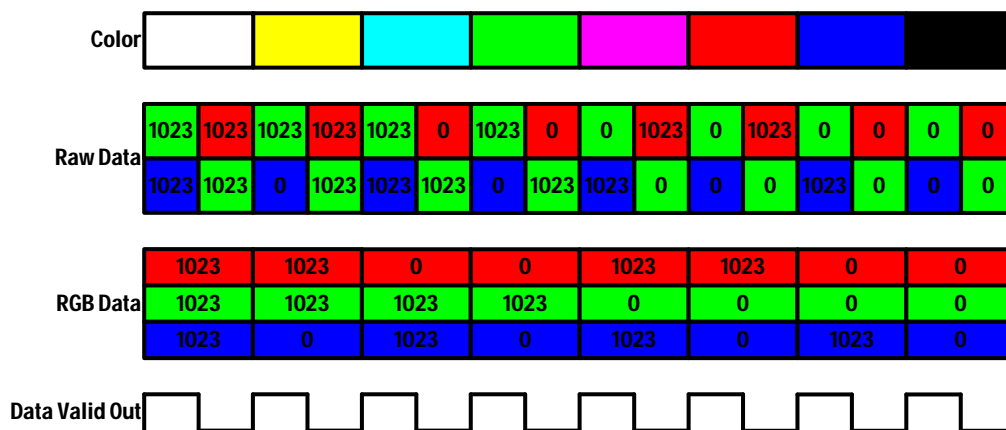


图 8 经过颜色插值后的预期输出

图 9为经过颜色插值后的实际输出，经过处理后的图像（每个像素包含R、G、B三种颜色分量）分辨率变为原来的一半，也就是说行列数都变为原来的二分之一。因此当待处理的图像分辨率为 640x480 时，经过颜色插值模块后，图像分辨率变为 320x240。通过与图 8经过颜色插值后的预期输出相比较可以发现，二者完全一致，说明该模块设计正确，达到了预期的目的。

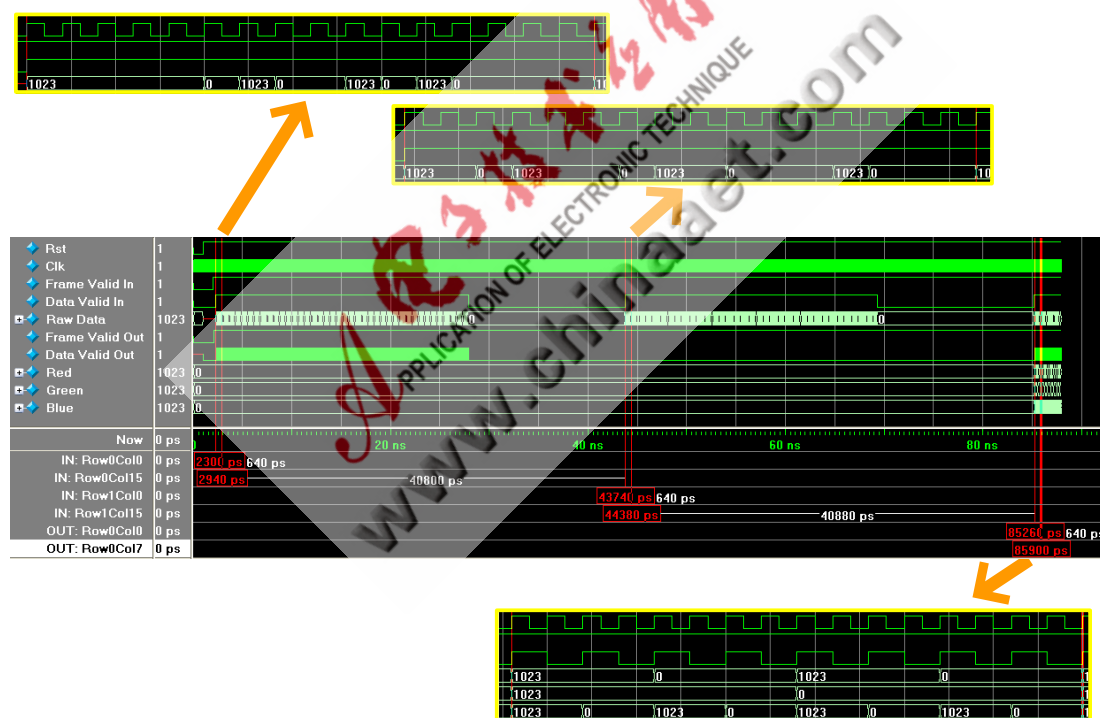


图 9 经过颜色插值后的实际输出

3.3 RGB2YCbCr 模块的设计

在可编程逻辑电路中，定点运算比较容易实现，因此用可编程逻辑器件实现颜色可见转换一般都采用定点运算。从RGB颜色空间到YCbCr颜色空间的转换公式如式(1)所示。转换矩阵的系数可以通过有限字长的定点无符号二进制数来表示。转换矩阵的系数原来都是范围从-1 到 1 的实数。采用宽度为n位的无符号二进制数表示转换系数的绝对值。设转换矩阵的某个系数为c，则其定点表示

为 $c*2^n$ 的四舍五入取整数。例如RGB颜色空间到YCbCr颜色空间的转换矩阵的第一行第一列为 0.257，采用 10 位无符号二进制来表示，其系数应该为 $0.257*1024=263.168$ 的四舍五入后取整，即为 263。

若采用 10 位来表示转换矩阵系数，得到 YCbCr 坐标与 RGB 坐标之间的关系如下：

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 263 & 516 & 100 \\ -152 & -298 & 450 \\ 450 & -377 & -73 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 66 \\ 512 \\ 512 \end{bmatrix} \quad (2)$$

这里R、G、B信号的范围是 0~1 的归一化信号。在实际电路中，R、G、B 信号由模拟信号通过AD转换得到，其信号宽度取决于AD芯片的位数。把宽度为 n 位的R、G、B信号记为 R_n 、 G_n 、 B_n 信号，其值的范围是 $0 \sim 2^n - 1$ 。例如 10 位的AD芯片，将得到 10 位的 R_{10} 、 G_{10} 、 B_{10} 信号，其值的范围是 $0 \sim 1023$ 。把式(2)乘以 2^n 得到：

$$\begin{bmatrix} Y*2^n \\ Cb*2^n \\ Cr*2^n \end{bmatrix} = \begin{bmatrix} 263 & 516 & 100 \\ -152 & -298 & 450 \\ 450 & -377 & -73 \end{bmatrix} \begin{bmatrix} R_n \\ G_n \\ B_n \end{bmatrix} + \begin{bmatrix} 66*2^n \\ 512*2^n \\ 512*2^n \end{bmatrix} \quad (3)$$

按照上式就可以直接利用 AD 量化后的 R_n 、 G_n 、 B_n 值来计算 $Y*2^n$ 、 $Cb*2^n$ 和 $Cr*2^n$ 。最后只要从得到的 $Y*2^n$ 、 $Cb*2^n$ 和 $Cr*2^n$ 取出有效数据位（输出有效数据位等于 n ）即可得到 Y、Cb、Cr。例如输出有效位数为 10 位，即可以通过取 $Y*2^n$ 、 $Cb*2^n$ 和 $Cr*2^n$ 的高 10 位来表示 Y、Cb、Cr。采用截断舍去无效位的方法的缺点是得到的 Y、Cb、Cr 总是偏小的。为了弥补这点可以通过增加平移来补偿。增加平移后的表达式如下：

$$\begin{bmatrix} Y*2^n \\ Cb*2^n \\ Cr*2^n \end{bmatrix} = \begin{bmatrix} 263 & 516 & 100 \\ -152 & -298 & 450 \\ 450 & -377 & -73 \end{bmatrix} \begin{bmatrix} R_n \\ G_n \\ B_n \end{bmatrix} + \begin{bmatrix} 66.5*2^n \\ 512.5*2^n \\ 512.5*2^n \end{bmatrix} \quad (4)$$

以输入为 10 位的 R_{10} 、 G_{10} 、 B_{10} 信号为例，系数采用 10 位二进制表示的颜色空间转换系统框图如图 10所示。为了提高该模块的性能，采用流水线结构实现此模块设计，在每个运算部件（包括乘法器和加减法器）的输出以及系统的输入输出之间加上寄存器缓存，减小了寄存器与寄存器之间的组合逻辑电路规模，从而大大提高了运算速度。

为了检验算法的正确性，把RAW2RGB模块输出的 8 种典型颜色的RGB数据作为颜色空间转换模块的输入，波形仿真如图 11所示。由波形图可以看到，相比于输入，输出结果延迟 5 个时钟周期出现，这是使用流水线结构造成的结果。虽然输出延迟了 5 个时钟周期，但每计算一个像素颜色转换仍只需要 1 个时钟周期。以输入 $(R, G, B) = (1023, 1023, 1023)$ 进行分析，在 5 个时钟周期后输出 $(Y, Cb, Cr) = (944, 514, 514)$ ，与图 12的预期输出相比较相差无几。其余实际输出数据与预期输出的比较详见图 12。

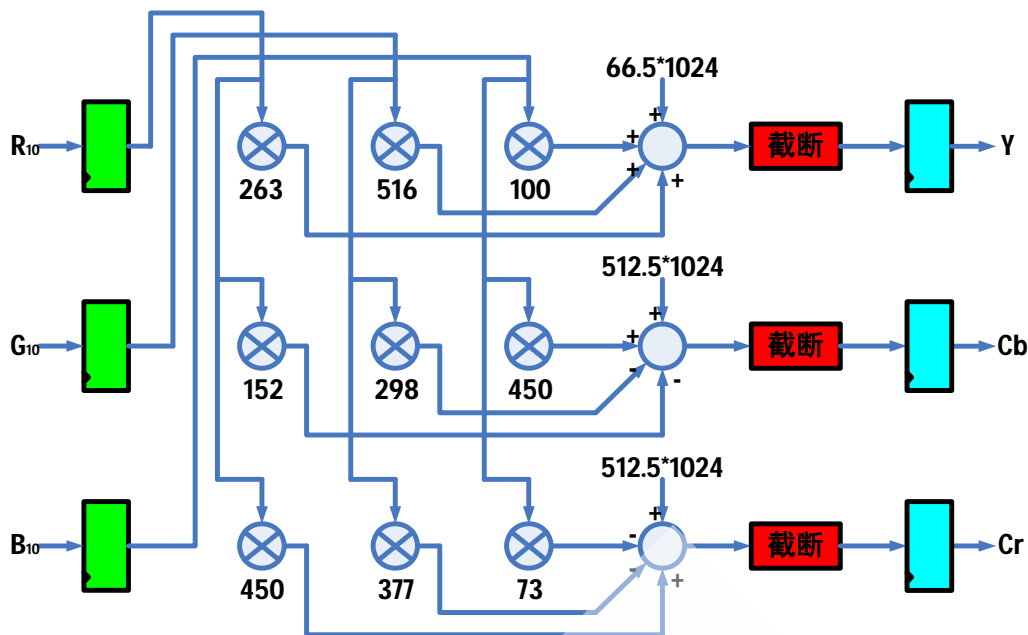


图 10 颜色空间转换的硬件结构框图

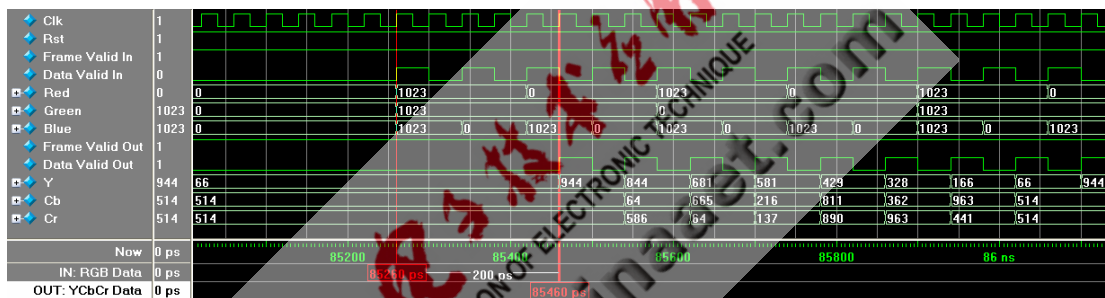


图 11 RGB2YCbCr 模块的仿真波形

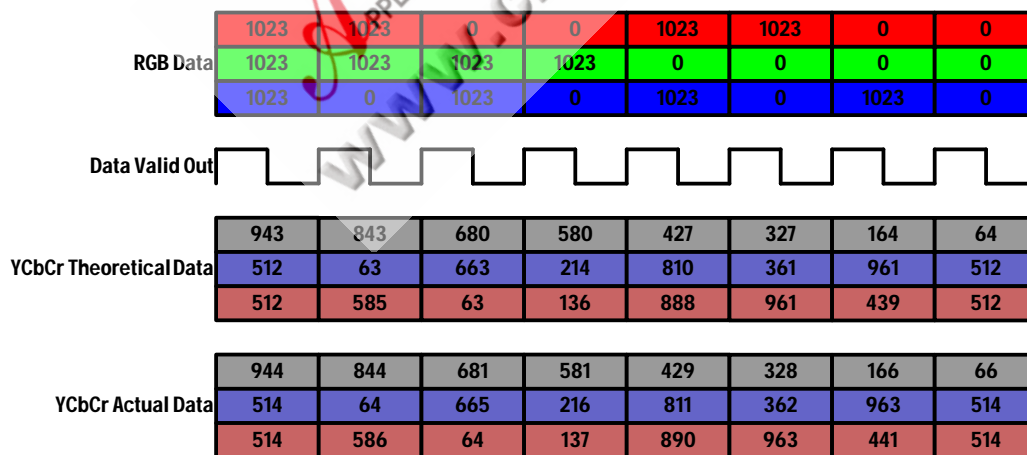


图 12 RGB2YCbCr 模块预期输出与仿真输出的比较

3.4 Otsu 模块的设计

3.4.1 类间方差表达式的简化

为了简化计算，改写类间方差的表达式，如式(5)所示。由于对于每帧图像来说， N 是一个固定的常数，可以忽略不计，所以式(5)可进一步改写为式(6)。

$$\begin{aligned}
\sigma^2 &= p_A(\omega_A - \omega_0)^2 + p_B(\omega_B - \omega_0)^2 \\
&= p_A p_B (\omega_A - \omega_B)^2 \\
&= \left(\sum_{i=0}^{\tau} \frac{n_i}{N} \right) * \left(\sum_{i=\tau+1}^{L-1} \frac{n_i}{N} \right) * \left(\frac{\sum_{i=0}^{\tau} i \frac{n_i}{N} - \frac{\sum_{i=\tau+1}^{L-1} i \frac{n_i}{N}}{\sum_{i=0}^{\tau} \frac{n_i}{N}}}{\sum_{i=\tau+1}^{L-1} \frac{n_i}{N}} \right)^2
\end{aligned} \tag{5}$$

$$\begin{aligned}
&= \frac{1}{N^2} \left(\sum_{i=0}^{\tau} n_i \right) * \left(\sum_{i=\tau+1}^{L-1} n_i \right) * \left(\frac{\sum_{i=0}^{\tau} i n_i - \frac{\sum_{i=\tau+1}^{L-1} i n_i}{\sum_{i=0}^{\tau} n_i}}{\sum_{i=\tau+1}^{L-1} n_i} \right)^2 \\
\sigma^2 &= \left(\sum_{i=0}^{\tau} n_i \right) * \left(\sum_{i=\tau+1}^{L-1} n_i \right) * \left(\frac{\sum_{i=0}^{\tau} i n_i - \frac{\sum_{i=\tau+1}^{L-1} i n_i}{\sum_{i=0}^{\tau} n_i}}{\sum_{i=\tau+1}^{L-1} n_i} \right)^2
\end{aligned} \tag{6}$$

观察式(6)， $\sum_{i=0}^{\tau} n_i$ 为累积直方图统计，而 $\sum_{i=\tau+1}^{L-1} n_i$ 可用 $\sum_{i=0}^{L-1} n_i - \sum_{i=0}^{\tau} n_i$ 来表示，

$\sum_{i=0}^{L-1} n_i - \sum_{i=0}^{\tau} n_i$ 可以通过累积直方图统计计算出来； $\sum_{i=0}^{\tau} i n_i$ 为累积灰度统计，而

$\sum_{i=\tau+1}^{L-1} i n_i$ 可用 $\sum_{i=0}^{L-1} i n_i - \sum_{i=0}^{\tau} i n_i$ 来表示， $\sum_{i=0}^{L-1} i n_i - \sum_{i=0}^{\tau} i n_i$ 也可以通过累积灰度统计计算出来。

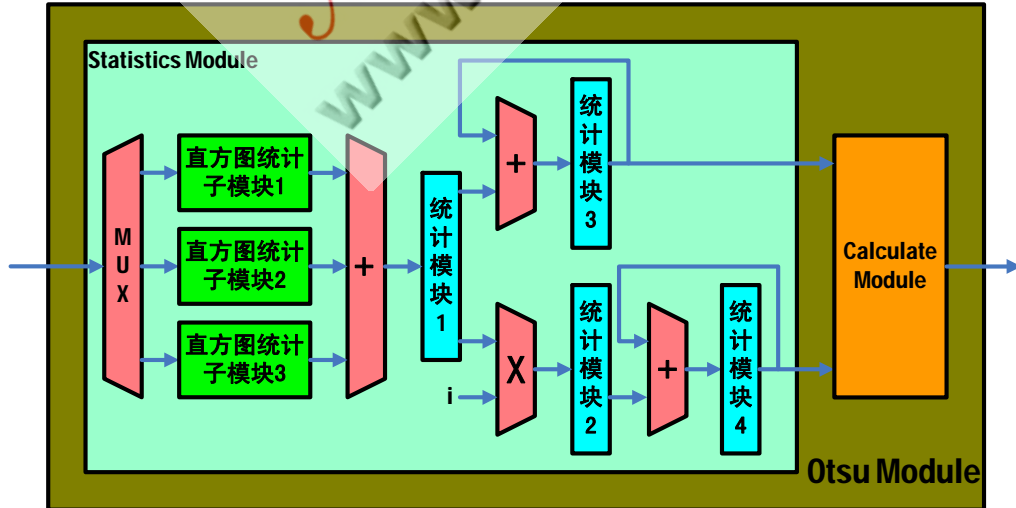


图 13 Otsu 模块的硬件结构框图

通过以上分析，可以把类间方差的计算分为两个步骤：首先是一帧图像的信息统计，包括直方图统计，灰度统计，累积直方图统计，累积灰度统计。然后按照式(6)的要求，依次把累积直方图和累积灰度统计中的数值送入计算模块

计算类间方差。Otsu模块的硬件结构框图如图 13所示，包含两个主要模块：统计模块与计算模块。

3.4.2 统计模块的设计

图像信息统计的核心是其直方图的统计，因为在直方图统计过程中存在循环传递路径，在读出某直方图元素的值后，对其加 1 再保存，而实际在循环中，下次要读出的直方图元素值可能是这次循环要写入的值，所以出现了一个循环传递依赖。因此，文献[4]提出的优化算法考虑采用冗余资源的方式解决，实现时在内存另外开辟 4 个临时直方图空间，用来作为保存中间结果的临时数组，每次取出 4 个图像像素，各点处理对应各自的中间直方图。统计完成之后，再归并 4 个中间数组得到最后的直方图。

用 FPGA 实现直方图的信息统计，也存在类似于软件的“循环传递依赖”问题，即在进行每一个像素统计时，都要完成读数、累加、写数 3 个步骤，然后才能统计下一个像素。三个步骤带来的处理延时降低了直方图统计的速度。

因此，我们在用FPGA进行直方图统计时，采用了三路并行直方图统计（即图 13中的直方图统计子模块 1、直方图统计子模块 2 和直方图统计子模块 3），和文献[4]采用的 4 个临时直方图空间统计具有异曲同工之妙。具体步骤如下：

(1) 在 FPGA 内构建常规的直方图统计模块。定义一个双口 RAM，将像素灰度值作为一个端口的读地址，读出 RAM 单元内容，将单元内容值加 1，然后将这个数从另一端口写入同一地址单元。

(2) 根据“以面积换速度”的思想，在 FPGA 内部构建三个相同的直方图统计模块，每个模块分别负责第 $3n$ 、 $3n+1$ 、 $3n+2$ 个像素。定义一个列计数器和一个简单的三状态状态机。行有效期间，在第 $3n$ 、 $3n+1$ 、 $3n+2$ 个时钟周期分别激活第一、二、三个直方图统计模块。这样，每个模块就有三个时钟周期来统计一个像素。从逻辑划分来说，读数、累加、写数三个部分的逻辑可以被划分为独立的时钟周期来完成，减少了同步寄存器之间的逻辑规模，提高了处理时钟频率。用这种方式，每个时钟周期可以统计一个像素。

(3) 在直方图输出时，将三个直方图统计模块的输出结果累加，就可以得到完整的直方图统计（即图 13中的统计模块 1）。此部分只需两个加法器就可完成。

得到完整的直方图后，可以通过状态机控制，依次得到灰度统计（即图 13中的统计模块 2），累积直方图统计（即图 13中的统计模块 3），累积灰度统计（即图 13中的统计模块 4），然后再把累积直方图统计和累积灰度统计中存储的数值按照式(6)的要求依次送入计算模块。

由于本系统传递到 Otsu 模块的图像分辨率为 320×240 ，每个像素采用 10 比特来表示，根据此时的数据量，可以很容易计算出每个统计模块的宽度和深度。为了便于今后程序的扩展（如图像分辨率的增加，单位像素比特位数的增加等），可以保留一定的裕量。这里为了方便起见，其中每个统计模块都采用宽度为 32 比特，深度为 1024 的双口 RAM 来实现。双口 RAM 可以直接调用 Altera 的 altsyncram IP 核实现。

3.4.3 计算模块的设计

对于计算模块，本设计采用两个除法器，三个乘法器来实现。为了获得最佳性能，除法器与乘法器的IP核可以分别调用Altera公司的lpm_divide和

lpm_mult来实现。为方便起见， $\sum_{i=0}^{\tau} n_i$ 用input1 表示， $\sum_{i=\tau+1}^{L-1} n_i$ 用input2 表示， $\sum_{i=0}^{\tau} in_i$

用input3 表示, $\sum_{i=\tau+1}^{L-1} in_i$ 用input4 表示, 则类间方差计算的硬件实现框图如图 14所示,

采用流水线作业, 只需要 5 个时钟周期, 就可以得到最终的类间方差。

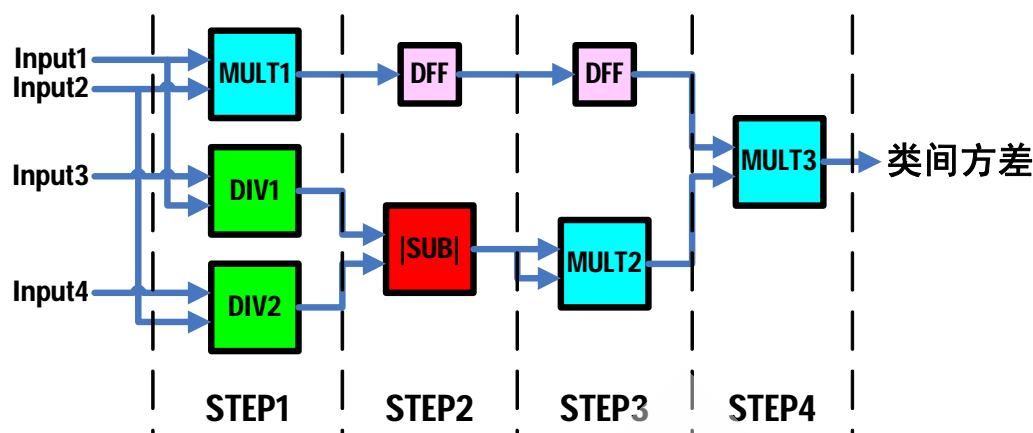


图 14 类间方差计算的硬件实现框图

3.5 Memory 模块的设计

Memory模块的设计如图 15所示, 接口模块负责处理SDRAM控制模块与本设计其余模块之间的数据交换, 由于是异步时钟接口电路, 所以采用异步FIFO作为数据交换的缓冲器。SDRAM控制模块用于产生使SDRAM按预期方式工作所需要的控制信号。

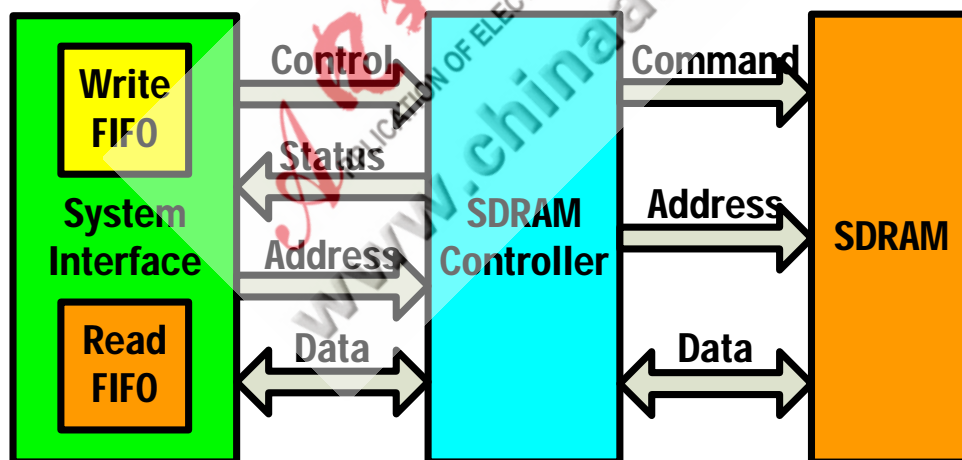


图 15 Memory 模块总体设计

3.5.1 SDRAM 控制模块的设计

SDRAM控制逻辑主要由 3 个模块组成: 主控制模块、信号产生模块和数据通路模块, 如图 16所示。主控制模块由两个状态机构成, 它是控制器设计的核心模块, 因为它会根据外部控制信号来控制SDRAM当前的工作状态; 信号产生模块则根据SDRAM的工作状态产生SDRAM需要的地址和控制信号; 而数据通路模块则是控制逻辑对SDRAM和对内的接口。

● SDRAM 的初始化

SDRAM在上电之后, 必须首先按照预定的方式进行初始化才能正常的运行。这里采用INIT_FSM状态机来处理其初始化, 其状态转换如图 17所示。

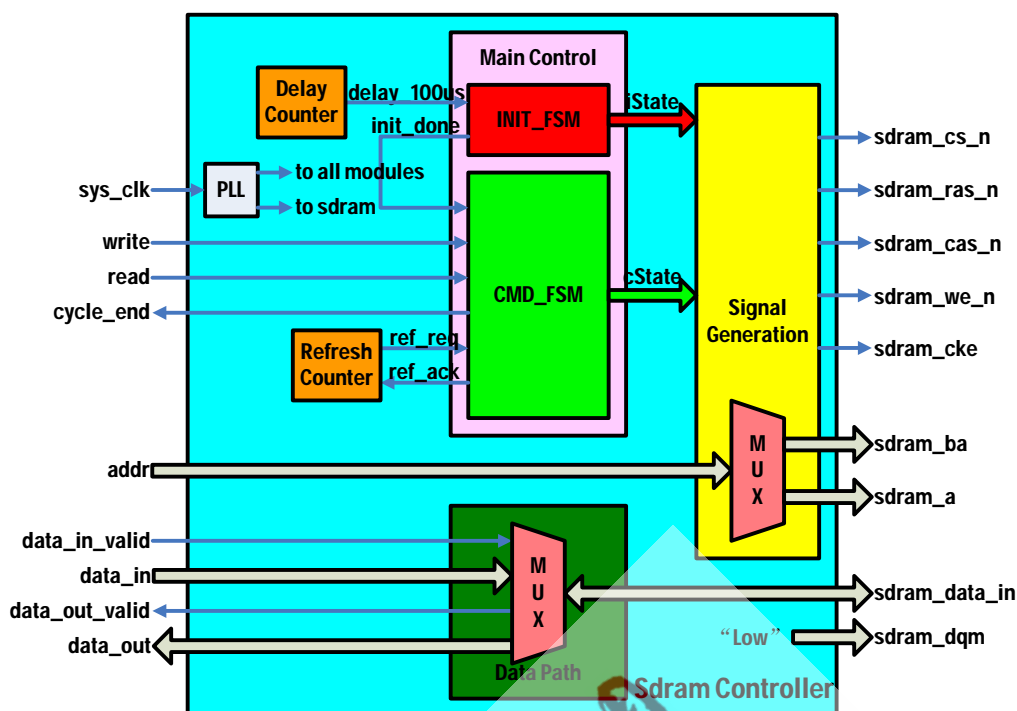


图 16 SDRAM Controller 模块

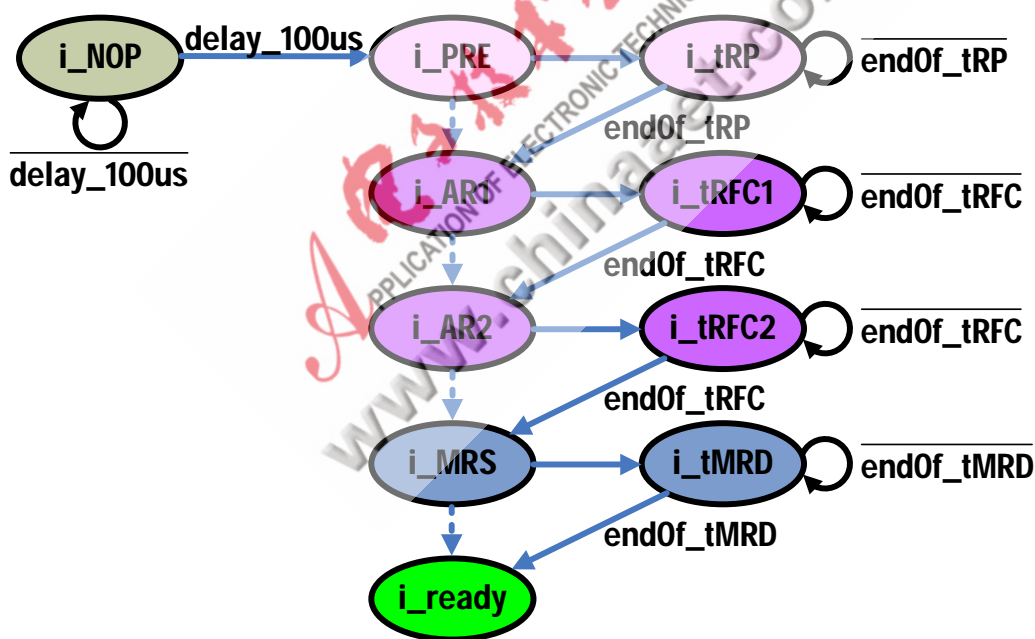


图 17 SDRAM 控制逻辑初始化状态图

SDRAM芯片的初始化仿真波形如图 18所示。仿真时调用的SDRAM模型是Micron公司的mt48lc1m16a1.v。观察仿真波形，芯片初始化的工作过程如下：

(1) 系统在上电后要等待 100μs，用于 power/clock 的建立和稳定，此时 INIT_FSM 状态机被强制在 i_NOP 状态；

(2) 对内存芯片执行 PRECHARGE 命令，完成预充电，INIT_FSM 状态机的状态由 i_NOP 转换为 i_PRE；

(3) 向内存芯片发出两条 AUTO REFRESH 命令，使 SDRAM 芯片内部的刷新计数器可以进入正常运行状态，INIT_FSM 状态机的状态分别为 i_AR1 和

i_AR2;

(4) 执行 LOAD MODE REGISTER 命令,完成对 SDRAM 工作模式的设定, INIT_FSM 状态机的状态由 i_AR2 转换为 i_MRS;

(5) 在 PRECHARGE、两条 AUTO REFRESH 和 LOAD MODE REGISTER 命令之间分别有延时 tRP、tRFC1、tRFC2 和 tMRD, 在延时 (tRP、tRFC1、tRFC2 和 tMRD) 被满足之前, SDRAM 只能执行空指令(NOP)。在执行完 LOAD MODE REGISTER 命令并等待延时 tMRD 后, INIT_FSM 状态机进入 i_ready 状态, 并在没有任何读写的情況下一直保持该状态, 这时 SDRAM 可以正常工作, 等待控制器对其进行读、写和刷新等操作。

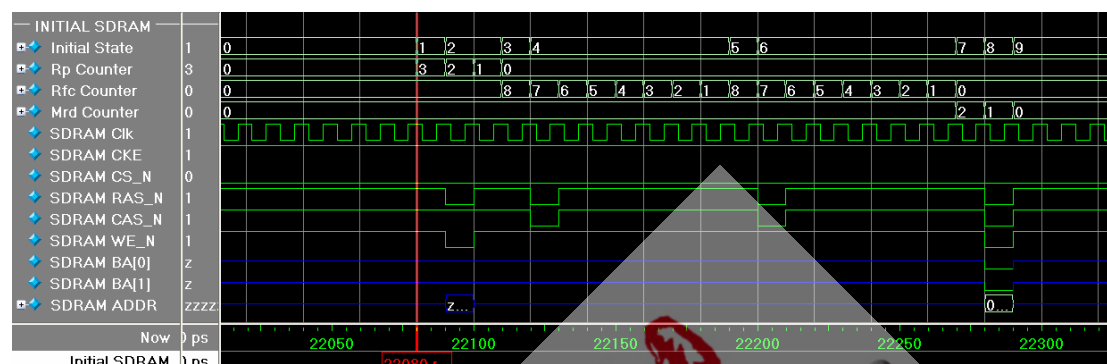


图 18 初始化 SDRAM 仿真波形

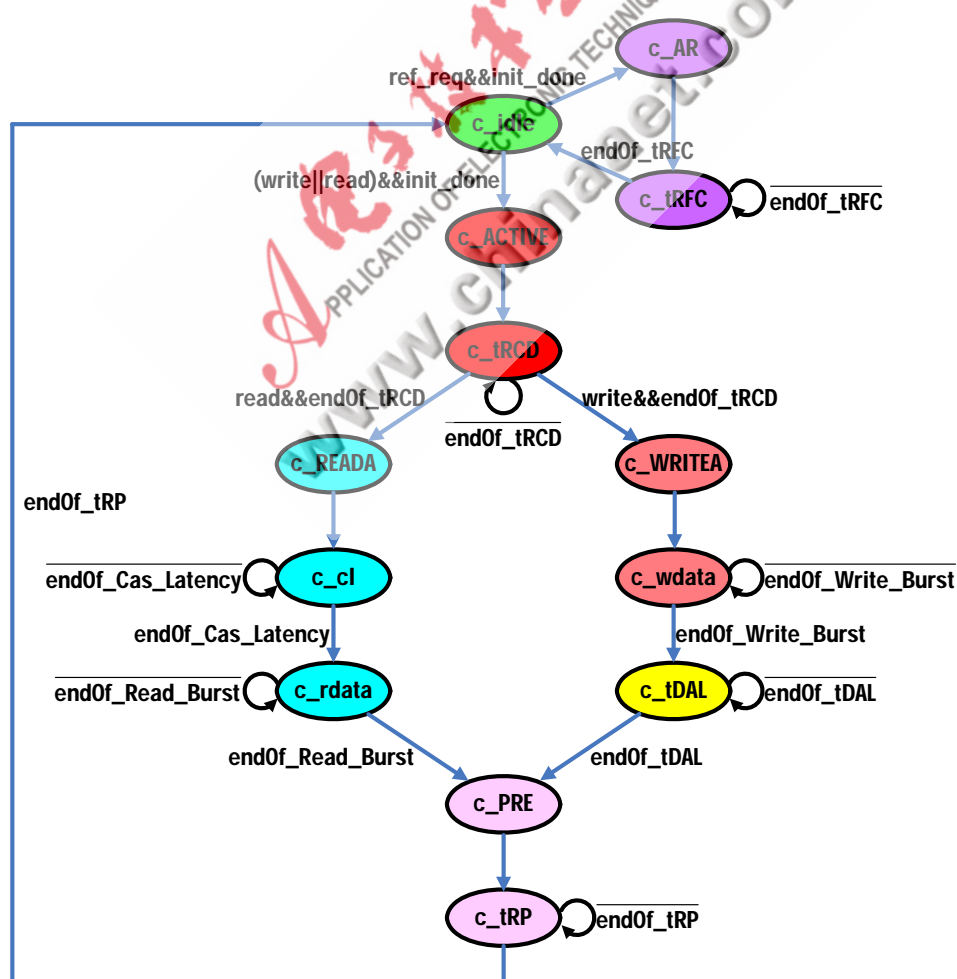


图 19 SDRAM 控制器工作状态图

● 读写操作

该 SDRAM 内部存储单元由 4 个 Bank 组成。每个 Bank 相当于一个矩阵，而行地址就是矩阵的行元素，列地址相当于矩阵的列元素。通过对行地址和列地址的选取，就能唯一的指定一个存取单元，当然还需要提供 Bank 地址。

SDRAM 的所有行在初始化后都处于非活动状态，所以必须激活相应的行，这就是 ACTIVE 状态存在的原因。在 ACTIVE 命令执行完毕，并经过 t_{RCD} （从行地址选择到列地址选择的延迟）的延迟之后，就可以根据 read/write 信号决定是读操作还是写操作。

本设计中的 SDRAM 工作在突发模式状态下，每次读写的突发长度为 8，所以在每次突发读/写完毕后状态机都会自动强制进入预充电状态，相应行也会关闭，以便下次能够正确读写该地址的数据。其状态转换如图 19。由于 SDRAM 的读写操作与接口模块的设计联系紧密，该部分的仿真波形在接口模块中分析。

● 定时刷新

由于 SDRAM 存储器是使用电容作为数据的存储装置，所以必须定时刷新，以保持原有的数据不丢失。一般认为，SDRAM 可以在 64ms 内保持数据的不丢失。所以，必须在 64ms 之内对 SDRAM 刷新一次。而 SDRAM 每次只能刷新一行，也就是说，若 SDRAM 芯片有 $2^{12}=4096$ 行，那每隔 $64ms/4096=15.625\mu s$ 必须刷新一次。刷新信号由 ref_req 产生，SDRAM 控制器输出 ref_ack 来通知系统 SDRAM 正在进行刷新，因为在刷新期间不允许系统对其进行读/写操作。

在接收到 ref_req 的刷新请求后，CMD_FSM 状态机进入 c_AR 状态，执行 AUTO REFRESH 命令。在时延 t_{RFC} 被满足后，CMD_FSM 状态机返回至 c_idle 状态。其状态转换如图 19 所示。

3.5.2 接口模块的设计

接口模块的结构框图如图 20 所示，该模块的主要任务是协调两块 Bank 的读写操作。在状态 1 时，FPGA 通过写 FIFO 缓冲，将采集的一帧图像数据保存到 Bank0 中，同时读出 Bank1 中的图像数据通过读 FIFO 的缓冲送入比较模块。当下一帧图像到来时，两块 Bank 通过 Switch Bank 信号进行总线切换，进入状态 2。Bank0 中的数据供后端读出送入比较模块，Bank1 开始存储下一帧图像数据，如此循环，完成数据的无缝缓冲与处理。在本设计中，Switch Bank 信号为帧有效信号^[5]。为了缩短设计时间并获得较佳性能，FIFO 可以直接调用 Altera 的 dcfifo IP 核实现。读/写 FIFO 的深度设置为 512，宽度设置为 16 比特。

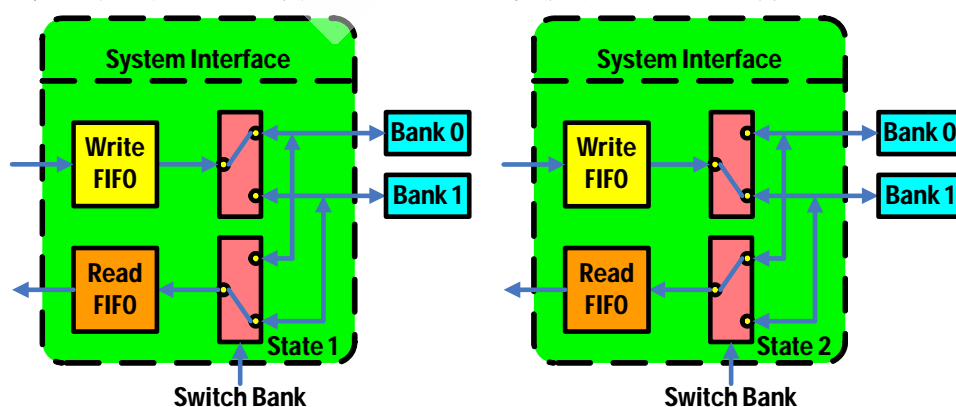
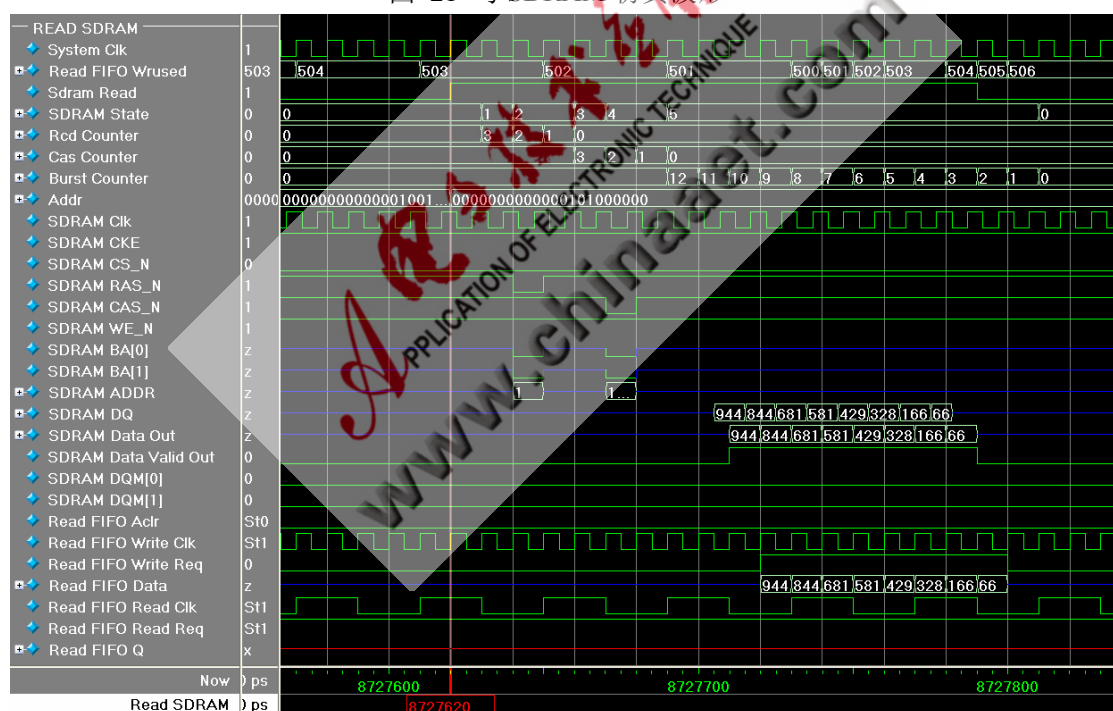
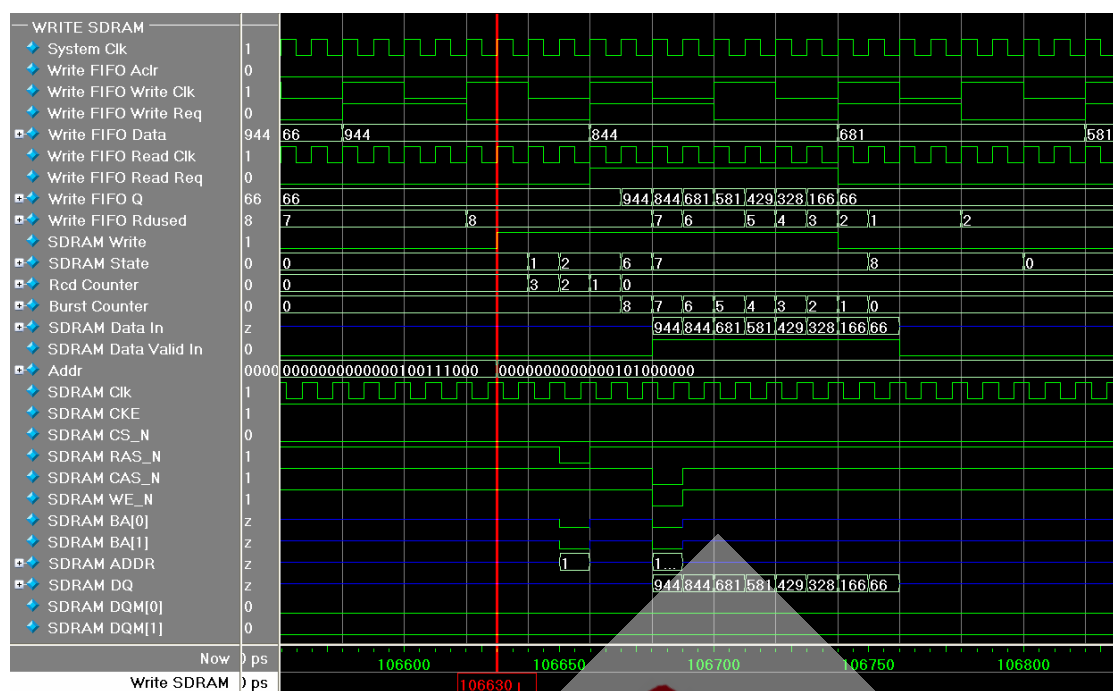


图 20 SDRAM 芯片与系统的接口设计



写SDRAM的仿真结果如图 21所示，读SDRAM的仿真结果如图 22所示。观察仿真波形，接口模块的工作过程如下：当写FIFO中存储的数据个数大于等于 8 时（因为SDRAM的写突发长度为 8，即一次要从写FIFO中读出 8 个数据），System Interface模块发出写SDRAM指令，随后SDRAM控制器根据写SDRAM指令控制状态转移，同时写FIFO要根据SDRAM控制器反馈回来的信号适时的从写FIFO中读出 8 个数据，以便完成SDRAM的写操作。同样的，当读FIFO中存储的数据个数小于 504 时（因为SDRAM的读突发长度为 8，即一次要往读FIFO中写入 8 个数据），System Interface模块发出读SDRAM指令，随后SDRAM控

制器根据读SDRAM指令控制状态转移，以便完成SDRAM的读操作，同时读FIFO要根据SDRAM控制器的输出数据及输出数据有效信号适时的把读出的 8 个数据写入到读FIFO中。

3.6 VGA 时序控制模块

VGA 时序控制模块是整个显示控制器的关键部分，最终的输出信号行、场同步信号必须严格按照 VGA 时序标准产生相应的脉冲信号。对于普通的对于普通的 VGA 显示器，其引出线共含 5 个信号：G、R、B（3 基色信号）、HS（行同步信号）、VS（场同步信号）。在 5 个信号时序驱动时，VGA 显示器要严格遵循“VGA 工业标准”，即 640Hz×480Hz×60Hz 模式。通常用的显示器都满足工业标准，因此设计 VGA 控制器时要参考显示器的技术规格。

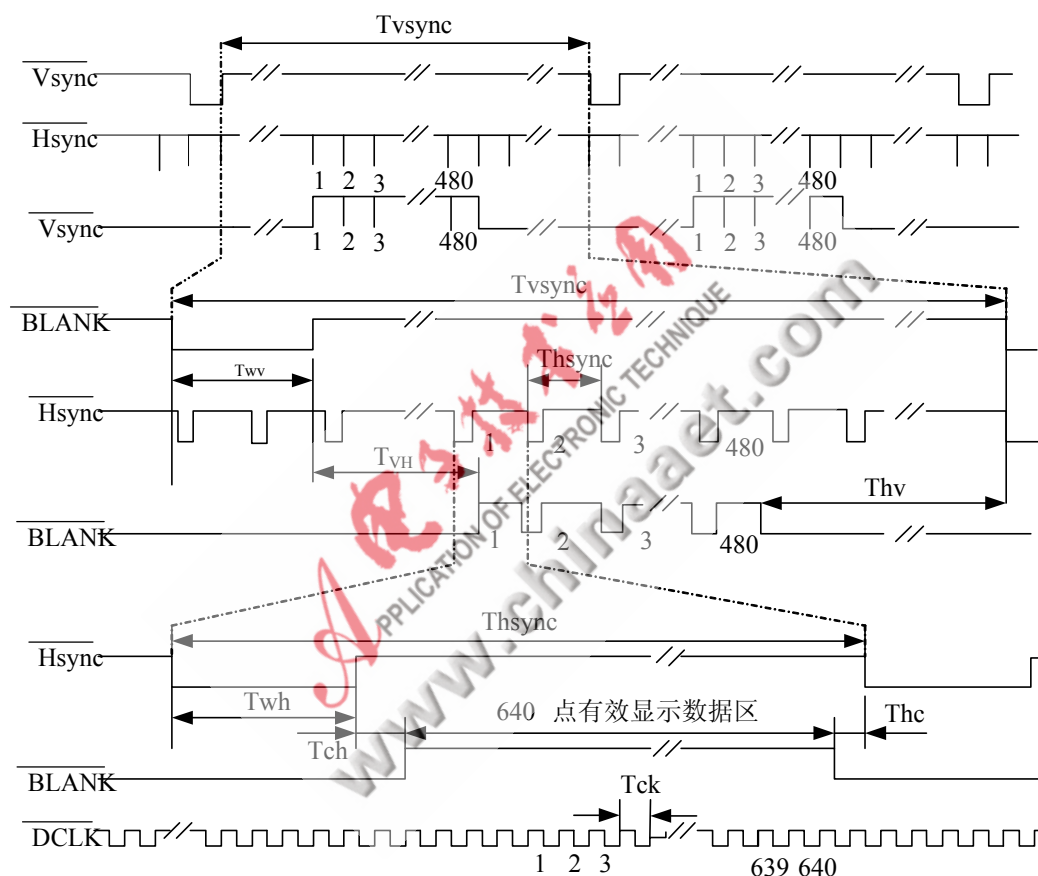


图 23 VGA 行扫描、场扫描的时序图

图 23所示是计算机VGA（640×480，60Hz）图像格式的信号时序图，其点时钟DCLK为 25.175MHz，场频为 59.94Hz。图中，Vsync为场同步信号，场周期Tvsync为 16.683ms，每场有 525 行，其中 480 行为有效显示行，45 行为场消隐期。场同步信号Vs每场有一个脉冲，该脉冲的低电平宽度 t_{wv} 为 63 μ s（2 行）。场消隐期包括场同步时间 t_{wv} 、场消隐前肩 t_{HV} （13 行）、场消隐后肩 t_{VH} （30 行），共 45 行。行周期 T_{HSYNC} 为 31.78 μ s，每显示行包括 800 点，其中 640 点为有效显示区，160 点为行消隐期（非显示区）。行同步信号Hs每行有一个脉冲，该脉冲的低电平宽度 t_{WH} 为 3.81 μ s（即 96 个DCLK）；行消隐期包括行同步时间 t_{WH} ，行消隐前肩 t_{HC} （19 个DCLK）和行消隐后肩 t_{CH} （45 个DCLK），共 160 个点时钟。复合消隐信号是行消隐信号和场消隐信号的逻辑与，在有效显示期复合消隐信号为

高电平，在非显示区域它是低电平^[6]。

为了便于比较结果，本设计把VGA显示器划分为4个区域，分别显示彩色图像，灰度图像和分割后的二值图像，每幅图像的分辨率都是320x240。剩下的最后一部分用于文字显示说明，最终的图像处理结果如图24所示。



图 24 图像处理结果

4 结束语

本设计采用 Altera 公司 Cyclone 系列的 FPGA 作为主控芯片，通过对图像传感器采集的图像数据进行存储，针对每一帧图像计算其最大类间方差，从而得到阈值，实现了图像的最大类间方差分割，并把分割后的图像通过 VGA 显示。

本设计的主要特色有以下几点：

- (1) 本系统具有图像静止功能，用户可以通过按键控制画面的静止和运动。
- (2) 本系统可以通过 LED 显示当前帧的阈值以及视频捕捉的帧数。
- (3) 本系统可以通过拨码开关设置 MT9M011 的寄存器参数（曝光时间，行/场消隐周期，彩色充盈度等），然后利用 I2C 模块对图像传感器进行配置，具有较强的适应环境的能力。
- (4) 本系统采用了可编程逻辑器件 FPGA 作为主控芯片，将结构简单，计算量大的算法从 DSP 中分离出来，保证 DSP 有足够的时间完成目标识别、跟踪等其它任务。另外由于 FPGA 的可编程性，使整个系统具有较大的灵活性。

为了进一步提高图像的分割质量，下一步工作可以利用二值形态学中的闭操作对分割后的二值图像进行处理，从而降低噪声的影响。

参考文献

- [1] 罗西平, 田捷. 图像分割方法综述[J]. 模式识别与人工智能, 1999,9(3): 300-312.
- [2] Otsu N. A Threshold Selection Method from Gray Level Histogram[J]. IEEE Trans. on Syst. Man, Cybern., 1979,9(1): 62-66.
- [3] Kittler J, Illingworth J. Minimum Error Thresholding[J]. Pattern Recognition, 1986,19(1): 41-47.
- [4] 雷涛. 多 dsp 并行多模跟踪软件系统的研究与优化实现. Master' s thesis, 中国科学院光电技术研究所, 2006.
- [5] 朱耀东, 张焕春, 经亚枝. 基于 FPGA 的一种高速图形帧存设计[J]. 电子技术应用, 2003(2): 72~74
- [6] 董士海, 张倪, 肖磊等. EGA/VGA 程序员手册. 北京大学出版社, 1999: 389~394

原创性声明

本人郑重声明：此处提交的论文《基于最大类间方差法的图像分割系统的设计与实现》，是本人在导师指导下，在哈尔滨工业大学期间进行研究工作所取得的成果。据本人所知，论文中除已注明部分外不包含他人已发表或撰写过的研究成果。对本文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明。本声明的法律效果将完全由本人承担。

作者签字 王暕来

日期：2008 年 6 月 15 日