# 基于 Shared Memory 的多核算法处理系统及实现

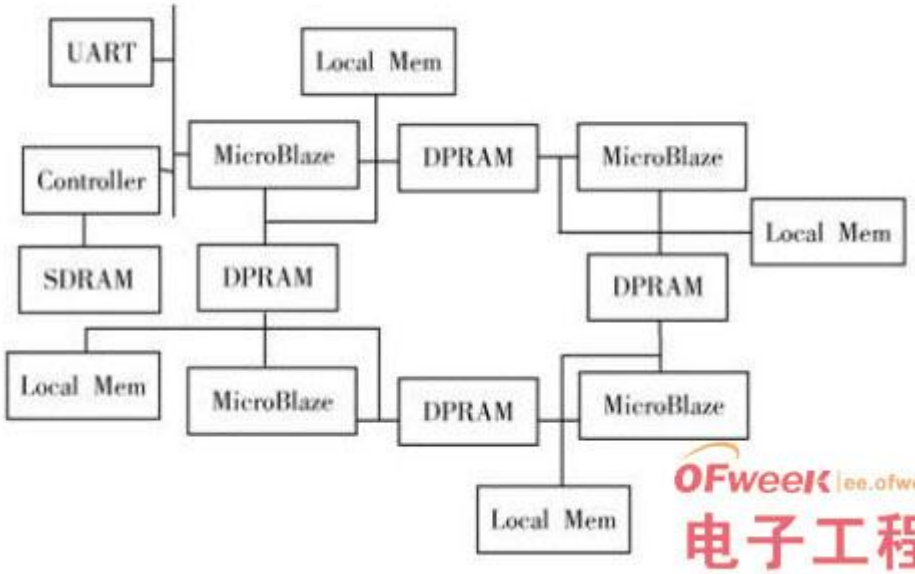
随着电子产业、工商业以及军事化产业的迅猛发展，越来越多的复杂运算已经无法单纯利用算法上的优化处理来大幅提升执行速度.为了解决日益复杂的计算问题,利用多核处理势必成为一种大势所趋.所以着手从多核和单核对比出发，利用 Xilinx 的 XUP Virtex-2 Pro 建立多核处理平台（基于 Shared Memory 通信机制）以及单核平台，并将相同的图像处理算法 DCT 分别运行在所构造的基于 FPGA 的单核和多核硬件平台上，观察实验结果，比较多核和单核运行所需的时间以及资源的消耗，最终的结果有力的说明多核在图像处理方面的绝对优势.

## 0 引言

基于 FPGA 的嵌入式应用在近几年来作为一个比较新颖的课题，已经在通信.消费电子.医疗.工业和军事等诸多领域占据了相当重要的地位.相对于其他芯片来说，使用 FPGA 设计的电路执行速度快.上市时间短.成本低廉.可靠性高以及易于维护升级.正是这些优点才使得 FPGA 的应用范围越来越广泛，备受各个领域设计师们的青睐.但是有关于它在多核体系上的研究却一直很少有人涉及.本文在研究各种核间通信机制的基础上，提出了一种基于 Mailbox 核间机制的多核处理系统，在该系统中集成了 Xilinx 的软核处理器 Microblaze，其降低了使用多信号处理板但来的成本问题同时还节省了空间，对更好的发挥多核系统提出了新的解决方案.

## 1 总线机制与核间通信机制

在多核嵌入式系统的设计中，核间通信机制与核间传输总线在选用时很有讲究，常用的总线有：OPB 总线.PLB 总线.XCL 总线.FSL 总线.LMB 总线，同时多核通信系统中常用的通信机制以及通信手段包括：Mailbox，Mu-tex，Shared Memory，Interrupt，PLBv46_PLBv46 Bridge，FSL 互连机制，DMA Controller 等.如图 1 所示.






### 1.1 PLB 总线

PLB 总线（Processor Local Bus）总线包括了一个总线控制单元.一个看门狗定时器以及独立的地址和读/写数据路径单元，另外，还包括了一个可选用的 DCR（Device Control Register）从接口以提供对总线错误状态寄存器的访问.

### 1.2 LMB 总线

LMB 总线主要用来连接片上 BRAM（BlockRAM）.为了能在一个时钟周期内完成访问，LMB 采用了最少的控制信号和简单协议的方式.它分为指令寄存器 DLMB 和数据寄存器 ILMB 两类接口，而且这些接口只和 BRAM 连接.

### 1.3 Shared Memory 通信机制

共享内存是一种典型的快速异步通信机制，因其使得零拷贝有可能实现，固非常适用于大于 1 000 B 的大型数据量共享的情况，共享内存可分为两种：BlockRAM 和外部内存 DDRR.

## 2 RGB2YCrCb 算法以及 DCT 算法介绍

RGB，YCrCb 是表示颜色时经常用到的两种颜色空间，在应用中经常需要实现它们之间的转换.例如在人脸检测中就常常用到 YCrCb 空间，因为一般的图像都是基于 RGB 空间的，在 RGB 空间里人脸的肤色受亮度影响相当大，所以肤色点很难从非肤色点中分离出来，也就是说在此空间经过处理后，肤色点是离散的点，中间嵌有很多非肤色，这为肤色区域标定（人脸标定.眼睛等）带来了难题.如果把 RGB 转为 YCrCb 空间的话，可以忽略 Y（亮度）的影响，因为该空间受亮度影响很小，肤色会产生很好的类聚.

而 DCT 变换是视频压缩编解码器中很重要的一部分，被广泛应用于各种视频格式的编码算法中，例如：

JPEG，MPEG1，MPEG2，H.264 等.

DCT 是先将整体图像分为 N×N 的像素块，然后对 N×N 的像素块逐一进行 DCT 变换.由于大多数图像的高频分量较小，对应于图像高频分量的系数经常为零，加上人眼对于高频成分的失真不太敏感，所以可以用更粗糙的量化.因此，传送变换系数的数码率要大大小于传送图像像素所用的数码率.图像到达接收端后通过反离散余弦变换回到样值，虽然会有一定的是真，但人眼是可以接受的，公式如下：

$$G(x,y,u,v)=\frac{2}{\sqrt{MN}}C(u)C(v)\cos\frac{(2x+1)u\pi}{2M}\cos\frac{(2y+1)v\pi}{2N}$$

式中 C（u） 和 C（v） 在 u.v 为 0 时等于 1 2 ，其他情况下均为 1，而 x，u = 0，1，2，…，M － 1 ； y，v = 0，1，2，…，N － 1.

### 3 多核系统设计环境与系统软硬件的设计

本文所采用的软件开发环境是 Xilinx 公司旗下的 ISE 10.1 开发套件，硬件开发平台采用的是 Xilinx 的 XUP Virtex-2 Pro[8-9]开发板，而 ISE 10.1 开发套件嵌入了 EDK 开发包（其集成了 Xilinx Platform studio，SoftwareDevelopment，库文件生成器，编译工具等开发模块），这样就大大方便了软硬件的开发.

### 3.1 利用 XPS 向导进行多核硬件系统设计

打开 XPS 软件后，首先利用 Base System Builder 向导建立一个单核系统，在此基础上通过添加新的 Micro-blaze 处理器软核以及必要的内存块.数据指令控制器和相应的外围设备来完成多核系统的创建.有关基于 BRAM 的共享内存机制已经简要的介绍过，这里不再重复.整个系统的框架如图 2 所示.

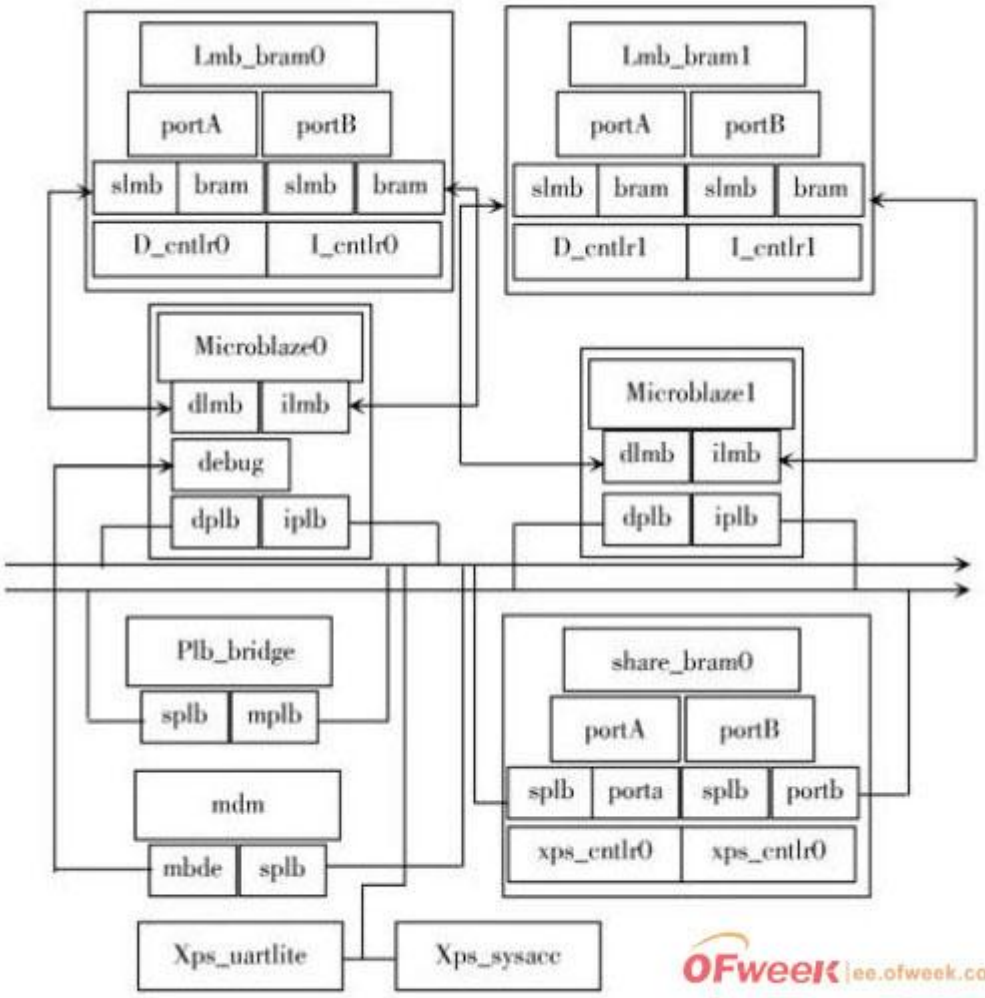





图 2　整个系统的框架

硬件系统采用的总线机制为 PLB 总线，所有的 Mi-croblaze 均挂在这两条总线上，并且以 Microblaze0 作为主处理器，其他用作从处理器.

设计中 PLB 总线的从设备主要是 xps-uartlite，它作为系统的主要验证手段，通过串口打印可以在 PC 机终端里输出核间通信信息.LMB 总线则用于连接片上 BRAM 和 MB 的各个 LMB 接口，实现了 D/I LMB 端口与启动内存块.数据/指令控制器的互连.

PLB 桥的作用是使得所有的处理器可以共享外围设备，如串口 RS 232 等.

### 3.2 利用 EDK 套件进行多核软件开发

在软件开发过程中，一方面除了需要注意重要的宏定义以及数据结构的设计外，如在 RGB2YCrCb 处理中，为了在后续的代码中对内存进行操作，需要对这些内存地址进行预先的宏定义以及设计有关 BMP 图片信息头的数据结构和信息获取函数；另一方面也要注意算法的设计.编译与部署.具体算法的核心部分设计如下所述.

RGB2YCrCb 的核心算法如下：

## 4 硬件实现

### 4.1 测试所搭建的硬件系统的步骤

（1）点击 Hardware 下的 Generate Bitstream 按钮，生成配置 FPGA 所需要的比特流文件；

（2）将 PC 机与 Virtex-2 Pro 开发板正确连接，并且给开发板上电；

（3）打开超级终端或是 Putty 工具，设置正确的参数（如波特率.奇偶校验位以及流控制位等），确保和创建硬件系统时的设置一致；

（4）点击 Device Configuration 下的 Download Bit-stream 进行比特流的下载配置；

（5）观察终端中的打印信息，判断是否和 Tes-tapp_Memory.c 文件中的一致.

### 4.2.1 图像算法在单核上的执行流程

（1）算法按照宏块顺序往下处理，先处理第一个宏块；

（2）对第一个宏块的前 4 个 Y 分量进行 RGB2YCrCb 处理，完成后对 YCrCb 结果做 DCT（YMatrix，color）变换；

（3）对第一个宏块的 Cr 分量进行 RGB2YCrCb 处理，完成后对 YCrCb 结果做 DCT（CrMatrix，color）变换；

（4）对第一个宏块的 Cb 分量进行 RGB2YCrCb 处理，完成后对 YCrCb 结果做 DCT（CbMatrix，color）变换；

（5）返回到步骤（1）进行第二个宏块的处理.

基于单核架构的处理流程是一个串行执行过程，体现在宏块于宏块之间是处于一种阻塞等待性的机制，换言之，只要宏块 0 的任何一个分量（Y/Cr/Cb）没有独立处理且最终完全处理完毕后处理器都必须等待，而无法转向宏块 1 进行后续的处理.

### 4.2.2 下载比特流，配置 FPGA

（1）正确连接好 Virtex-2 Pro 开发板，并给板子上电；

（2）打开一个超级终端或是 Putty 工具，注意设置匹配的波特率.奇偶校验位以及流控制位；

（3）对于需要运行的应用程序，右单击选中 Markto Initialize BRAMs 以初始化内存块；

（4）在 Device Configuration 中点击 Download Bit-stream 下载比特流，配置 FPGA；

（5）修改 shm.c 文件中 Trycount 值（该值为一幅 8 KB 图片进行循环处理的次数），然后重新编译.下载并配置 FPGA，统计每次处理完成的时间，并填写表 1.

表 1　单核处理平台的时间统计

| Trycount | 10 | 50 | 200 | 1 000 |
|---|---|---|---|---|
| 单核体系 /s | 6 | 29 | 116 | 598 |



### 4.3 多核执行图像处理算法以及时间测量

#### 4.3.1 图像算法在多核上的执行流程

（1）算法仍以宏块处理为单位，MB0 先对第一个宏块进行处理；

（2）MB0 对第一个宏块的 4 个 Y 分量进行 RGB2Y 变换，并将中间结果写入内存，作为 MB1 进行 DCT（YMa-trix，color）的输入数据；

（3）MB1 读取颜色转换后的 Y 分量，对其进行 DCT（YMatrix，color）；

（4）MB0 在将 Y 分量写入共享之后，立即开始 Cr 分量的 RGB2Cr 变换，并将 Cr 结果写入共享作为 MB1 进行 DCT（CrMatrix，color）变换的输入数据；

（5）MB1 读取颜色转换后的 Cr 分量，对其进行 DCT（CrMatrix，color）；

（6）MB0 在将 Cr 分量写入共享之后，立即开始 Cb 分量的 RGB2Cb 变换，并将 Cb 结果写入共享作为 MB1 进行 DCT（CbMatrix，color）变换的输入数据；

（7）MB1 读取颜色转换后的 Cb 分量，对其进行 DCT（CbMatrix，color）；

（8）MB0 完成一个宏块的处理之后不等待，即可返回步骤状态进行第二个宏块的处理，而 MB1 则以 MB0 的处理过程循环往复.

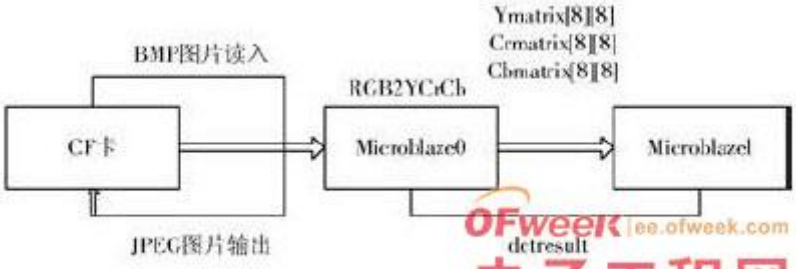




图 3　多核执行图像处理框架

基于双核架构的处理流程也是一个并行执行过程，体现在当 MB0 做好 RGB2YCrCb（所有分量的颜色变换）之后，只需将处理的中间结果写入内存作为 MB1 进行 DCT 变换的出入数据，而无需等待 MB1 上 DCT 处理进程的完成，在 MB1 进行上一个宏块的 DCT 变换处理过程中，MB0 已经转向宏块 1 进行下一轮的各个分量的 RGB2YCrCb 处理.

### 4.3.2 下载下载比特流，配置 FPGA

所有步骤同单核配置过程.运行后同样要改写 shm.c 文件中的 Trycount 值，记录时间，并填写表 2.

表 2　多核处理平台时间统计

| Trycount | 10 | 50 | 200 | 1 000 |
|---|---|---|---|---|
| 多核体系 /s | 2 | 3 | 12 | 47 |



通过表 1，表 2 的对比，很容易发现，多核体系在时间上有明显的优势.

### 4.4 单核/多核体系占用资源对比

通过编译，在生成的编译报告中将两者所用的资源统计如表 3 所示.从表中可以看到在硬件资源的占用上，双核体系的确消耗的资源较多.

表3 多核和单核处理平台各项资源对照表

| | 多核 | 单核 |
|---|---|---|
| Slice<br>寄存器总数 | 7 113(25%)<br>As Flip Flops:7112<br>As Latches:1 | 4 370(15%)<br>As Flip Flops:4369<br>As Latches:1 |
| 4输入的LUT查找表个数<br>(逻辑占用) | 8 977(32%) | 3 302(12%) |
| 被占用的<br>Slices个数 | 7 184(52%)<br>9 477(34%) | 4 068(29%)<br>3 468(12%) |
| 所有4输入的LUT<br>查找表总数<br>(分布式) | As logic:7 559<br>As a route-t hru:500<br>Dual Port RAMs:1152<br>As Shift registers:225<br>As 16*1 RAMs:41 | As logic:2 835<br>As a route-t hru:166<br>Dual Port RAMs:384<br>As Shift registers:83<br>As 16*1 RAMs:N/A |
| 绑定的I/D块数 | 32(5%) | 137(24%) |
| IOB触发器 | 60 | 444 |
| RAMB16使用个数 | 104(76%) | 41(30%) |
| 18×18乘法器的使用个数 | 9(6%) | 3(2%) |
| 缓冲区使用个数 | 3(18%) | 5(31%) |
| 数字时钟管理器使用个数 | 1(12%) | 2(25%) |
| BSCAN比特流下载线 | 1(100%) | 1(100%) |
| 管脚制定的I/O块个数 | 31(96%) | N/A |



## 5 结语

多核系统由于采用了并行环路体系并摒弃了单核阻塞状态下的等待时间，从而能够达到处理时间上的优化，但是另一方面其对片上资源的消耗也会随着从处理器的增加与日俱增，这也印证了“速度与面积”不能兼得这条原则，所以在工程中一定要做好权衡，权衡算法性能优化和硬件资源占用率的取舍，最好能够折中.

当然在处理如图像处理.信号处理类的一些复杂算法时，多核机制还是相当具有可取性的，毕竟在实际工程中，往往不介意以空间资源消耗来获取时间上的优化处理.