MobileNet\_VC709 Design specification

**v1.0 2021/8/19**  **spec writer 赵忠宇（澳门大学）**

设计目标

本项目目标为搭建一个部署在FPGA上的图像识别与分类系统，该系统能够读取摄像头或片外DRAM的图像信息，经过预处理后输入神经网络模型，完成inference操作后，通过HDMI协议将原图像与分类结果打印至显示器。

硬件平台

VC709 Connectivity Kit

· Virtex-7 VX690T

其资源总数见图[1].

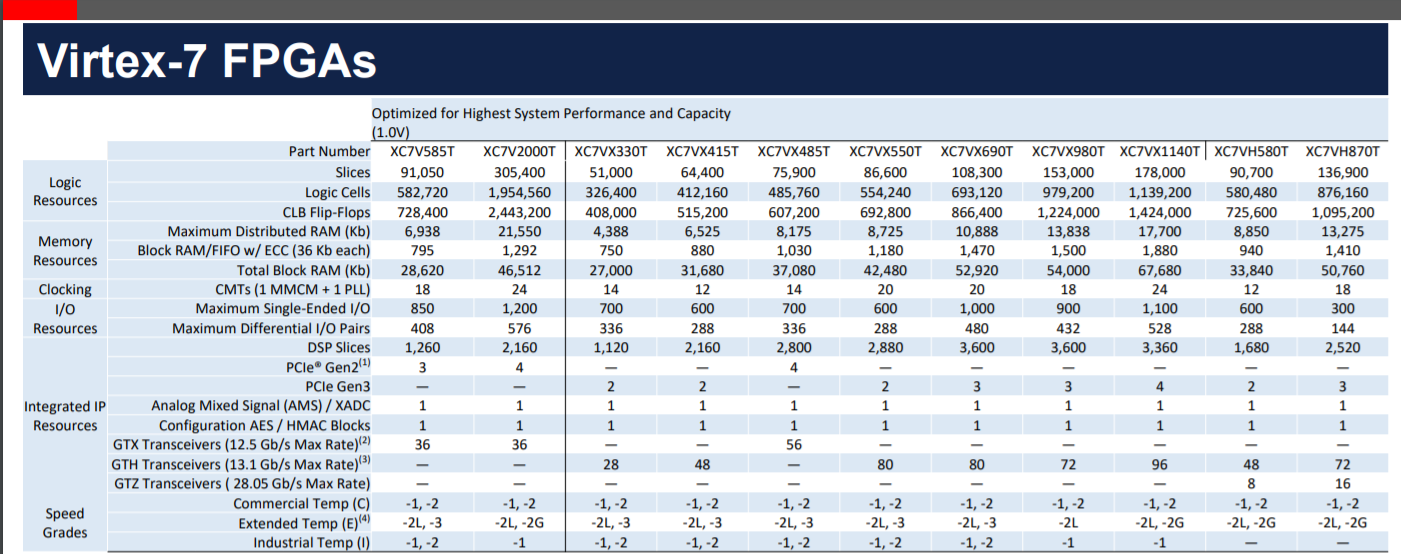


图 1 Virtex-7 系列芯片资源数目统计

1960\*1024液晶HDMI显示屏

Ov7670摄像头模块

功能分解

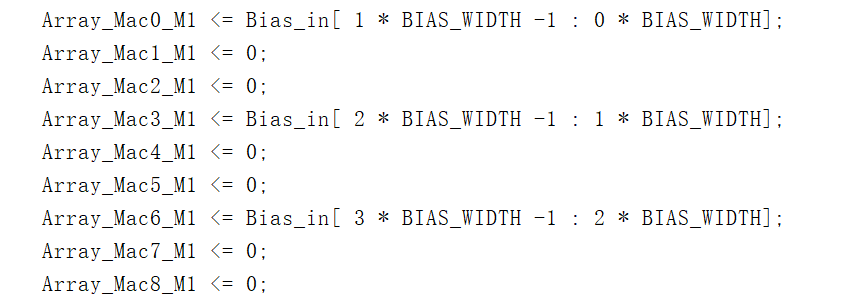
1. 系统在上电后，固化好的bit流能自动从Flash中下载至FPGA，对FPGA进行编程。
2. 系统分为DRAM工作模式与摄像头工作模式。DRAM工作模式下，系统从DRAM的指定位置，读取待inference的图片，以流的形式输入神经网络的输入层，之后启动神经网络，完成inference，将图片与结果共同显示在屏幕上。摄像头工作模式下，上电后系统对摄像头进行初始化，读取摄像头输入的视频流，之后分成两个线程，一个线程对输入帧进行预处理并以流的形式存入神经网络的输入层，神经网络处于连续inference模式，得出lable。另一个线程对视频流进行输出，lable的字符串像素与输出视频流进行merge后显示在屏幕上。
3. 神经网络的输入层为流输入形式。由于MobileNet第一层为普通3\*3卷积，故第一层NN\_front独立于其他层进行单独实现。该层的主要目的为将以流传输的数据流转化为以package存入BRAM中的数据流
4. MobileNet V2的主要部分为Inverse Residual Block（IRB），所有IRB层都由一个可配置的模块进行inference计算，该模块对输入值的内存读取模式与对输出值的内存写入模式相同，故可以在一个模块内连续进行多次的IRB层运算。该模块通过配置控制寄存器来实现对不同层的操作。
5. MobileNet V2的输出值经过softmax模块存入输出BRAM组中。该BRAM组将被HDMI显示模块调用，排序后将分类的前五名打印在屏幕上。

系统架构图

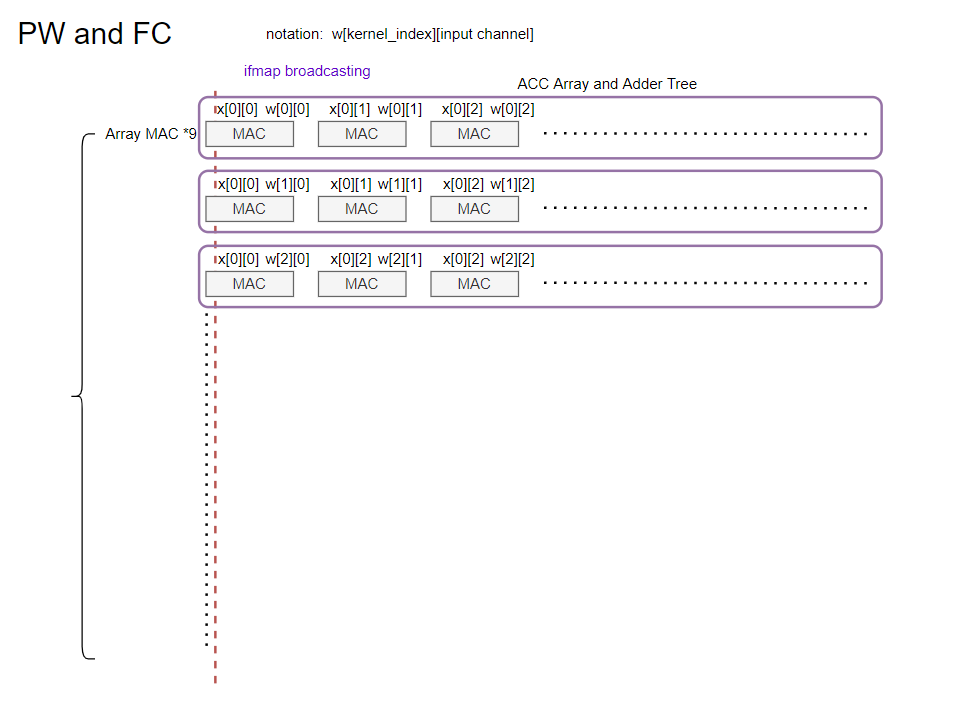
由于9个channel捆绑在一起，Ifmap channel与ofmap channel在配置时数值要除以9，并向上取整。

M1 Bram存储时注意计算第一层convolution时的存储方式：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **地址** | **寄存器名** |  | | | | | | | | | |
| **7** | **General\_config** | [31:22] ifmap\_channel | [21:11] ofmap\_channel | | [10:4] ifmap\_size | | **[2]stride** | | | [2:0]: Mode  00: Idle  01: Conv  02: DW  03: PW  04：AVGPOOL | |
| 6 | Ifmap\_base\_addr |  | | | | | | | | | |
| 5 | Ofmap\_base\_addr |  | | | | | | | | | |
| 4 | Weight\_base\_addr |  | | | | | | | | | |
| 3 | M1\_base\_addr |  | | | | | | | | | |
| 2 | M | [31:16] SC\_M2 | | | | [15:0] M0 | | | | | |
| 1 | SC\_base\_addr | [19:1] SC\_base | | | | | | | [0]: has\_SC | | |
| 0 | Zero\_points | [31] : ReLU\_enable | | [23:16] SC\_zero\_point | | | | [15:8] Output\_zero\_point | | | [7:0] Input\_zero\_point |



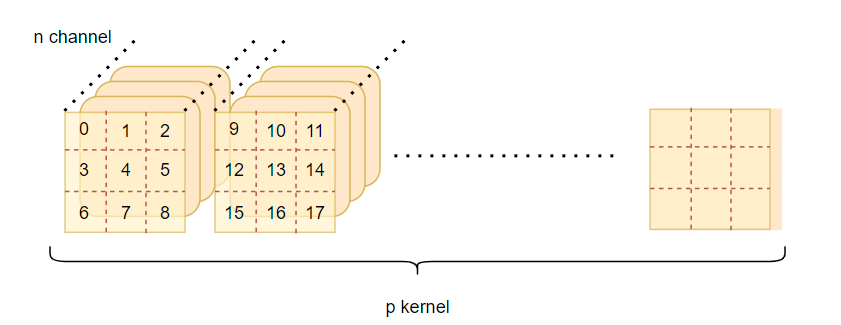
若采用9个ofmap\_channel捆绑至一个深度进行存储，在计算第一层普通卷积层时，当前kernel的index（p）值为4，而ofmap\_channel为11，二者之间存在3倍的关系，而在计算DW与PW层时，ofmap\_channel与kernel\_p是一样的。



Pointwise与Fully Connected层中kernel如何存储

在CONV与DW中，kernel的读取相对简单，因为3\*3的kernel正好让每一个Bram存储一个kernel的像素，但对于Pointwise与Fully Connected的kernel，如果要实现上图中的数据流，其存储方式就需要另作安排。

很显然，依照上图的数据流，在这两层中，不仅9个channel需要捆绑在一起，9个kernel此时也需要捆绑在一起同时输出了。这个时候就需要将所有kernel想象为一个feature map的排布，每次输出3\*3的一个block，直到所有的kernel输出完毕为止。



给定kernel的数目为p，channel数为n。则kernel的在BRAM中的地址可分为Kernel\_block\_subaddress与Channel\_subaddress，Kernel\_block\_subaddress值为p/9 （若有余数则加1），Channel\_subaddress值为n/9（若有余数则加1），

Average Pooling 实现方案：

Average Pooling 地址访问方式不同于前面任意一种方法：所有BRAM的读地址都一样，利用9个clk读取出一副frame，他们的值将在MAC中乘以一个定值，尔后在ReLU层进行移位，如此便可实现Average Pooling：

读数时，设置ifmap\_size为3，每次读出一个3\*3block，判断IHB与IVB的值即可对边缘点进行padding。而tlast\_no\_sync的值即等于IHB与IVB计满时。这样一来，IHB与IVB的计数条件可需要进行相应变换，即IHB随1clk自增，而仅有IHB与IVB增满后才使Channel\_cnt自增，为了不继续增加复杂度，设置两个新的计数器AIHB与AIVB，用于Average Pooling的计数。

将该层的M0设置为1337

Short Cut实现方案

带有Short Cut的PW2层需要以下参数

* 本一级PW2层的scale： , 本一级PW2层的zeropoint： , 本一级PW2层的输出
* 上一级PW2层的scale： , 上一级PW2层的zeropoint： , 上一级PW2层的输出
* ShortCut层的scale： , ShortCut层的zeropoint：

,

记ShortCut层输出为，另记 , , ,

故有式：

在配置阶段，需将相关参数全部配置于寄存器中，方便模块调取。在PW计算的过程中，包含Shortcut的PW层在收到write\_valid后，并不直接写入BRAM，而是首先进行寄存，同时BRAM写入的地址用于读取前一级的PW层的输出。

这意味着在包含Shortcut的PW层中，读取分为两个阶段，第一阶段陆续读出ifmap当前point的各个channel，

第二阶段读取用于Shortcut的前一级ofmap的channel。读取完毕后，可指示Shortcut\_done信号，继续重读ifmap当前point的各个channel，kernel\_cnt加1。

问题是这样做会中断原来PW层计算的流水线吗？在不包含Shortcut的PW层中，一旦收到tlast信号后，表明当前point的一个ofmap channel计算结束，ACC寄存器清零后随即可以开始下一个ofmap channel的运算。但包含SC的时候，下一个地址流中的1个clk需用于读取前一级的ofmap channel，这使得流水线需中断一个clk。可以在配置时，将含有shortcut的PW层的channel数目多配置一位。在读到这个位的时候，Data\_DMA切换base\_address，读前一级的ofmap，如此一来，便实现了前一级参与本一级的运算。