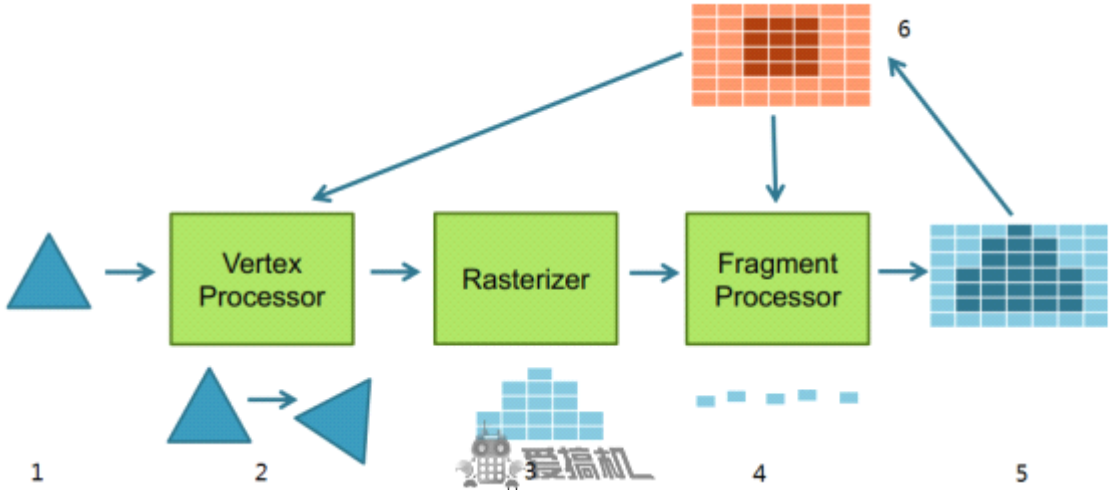
前言

现在移动设备的“核战”越来越激 烈，已经从CPU引发到了GPU上，于是“16核”、“8管线”、“MP4”、“三角形生成率”和“填充率”等各种吸引眼球的宣传铺天盖地而来。一直很希 望能有些文章来介绍科普下，但或许是专业人士都觉得这些太基础，最后忍不住，只能由我这个半桶水的非专业人士来写点啥了。本文参考和引用了网上的一些资 料，篇幅有限恕不一一列举。内容尽可能的科普浅显，但限于个人的知识层次、理解能力和表达能力，如果有不确切或者错误的地方，还请多多指正。

基本的3D流水线

首先我们来简单的介绍下3D的画面是如何生成的，一个基本的3D流水线如下图所示：

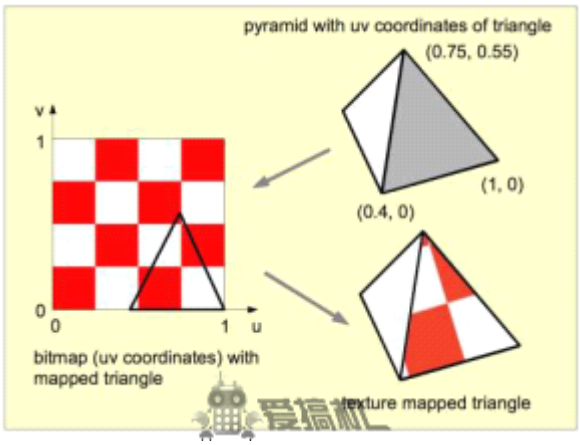


首先，运行在CPU上游戏引擎根据游戏中的一些参数，产生一系列的图元，将它们的顶点数据发送给GPU。

第二步，顶点处理器（Vertex Processor）对顶点数据进行一系列的变换和光照处理。简单的想想，游戏中的各个物体的坐标都是参照游戏中的世界坐标系的，而实际显示的画面是玩家 视角或者摄像机视角，这中间就有许多坐标系的转换。这些活就需要顶点处理器来做，最终我们得到了我们所需要视角的画面。

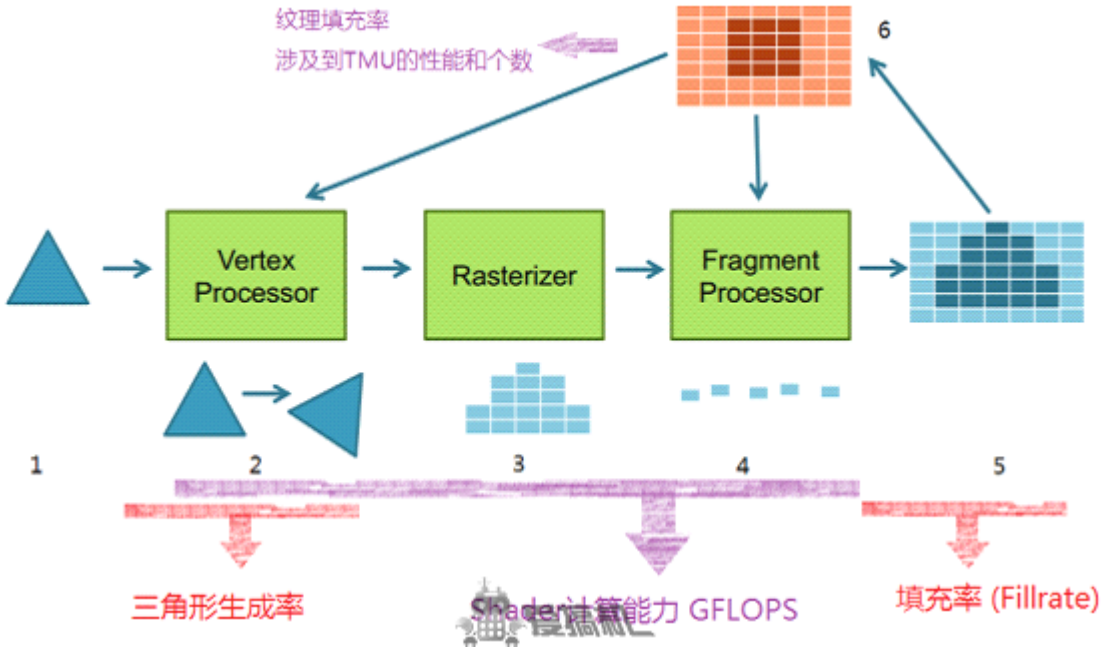
但是，到这一步，画面还只是一些多边形，而实际显示在屏幕上的是一个个像素，这里就需要Rasterizer进行光栅化（Rasterization，3），从而将画面变成一个像素图。

第四步，对这些像素进行上色，Fragment Processor中的像素着色器（Pixel Shader）按照程序规定的算法，计算出画面中每个像素的颜色，之后第五步就是把结果输出到内存中，全部完成后拿去显示。当然在这整个过程中，还跟纹理 贴图6有关系。所谓贴图就是把纹理（一个二维的静态图片）按照一定的算法给贴到游戏里的三角形表面上去。



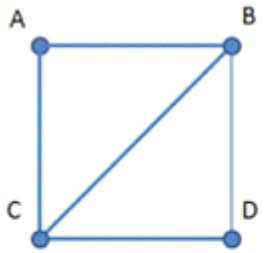
参数，究竟靠不靠谱？

终端厂商在宣传他们手机芯片的 GPU 如何强劲时，往往会提到一些参数，最常见的就是三角形生成率和填充率。**但实际上，不同公司的 GPU，他们的这些理论参数，并不具有直接的可比性。**我 们也可以发现，有些 GPU 可能给出的理论参数很高，但实际表现却很一般，甚至不如一些参数低的 GPU。这是因为，各个 GPU 的供应商，比如 IMGtec（PowerVR SGX），高通（Adreno），Vivante（GC 系列），ARM（Mali），nVIDIA（GeForce），他们给出这些理论数据的测试方式， 可能不太一样。



比如三角形生成率，本身就受到很多测试因素的影响。例如，有些三角 形在一开始就会被剔除掉（比如在屏幕外，或者太小了根本覆盖不到一个像素），不会被显示也不会需要执行太多的运算，那么这部分三角形到底应该算吗？如果算 进去的话，三角形生成率自然就高了。或者，测试程序把一些已经计算好的坐标提交给 GPU，那么 GPU 的 Vertex Shader 就不需要进行复杂的计算，数值自然就高了。再或者，并非生成一个三角形就一定需要处理三个顶点。如果有三角形共享顶点，用 indexed 的办 法绘制，顶点的数目就可以减少。就像下

图，2 个三角形只需要处理 4 个顶点。如果测试的时候大量使用这种方式，也可以提高三角形生成率的数值。



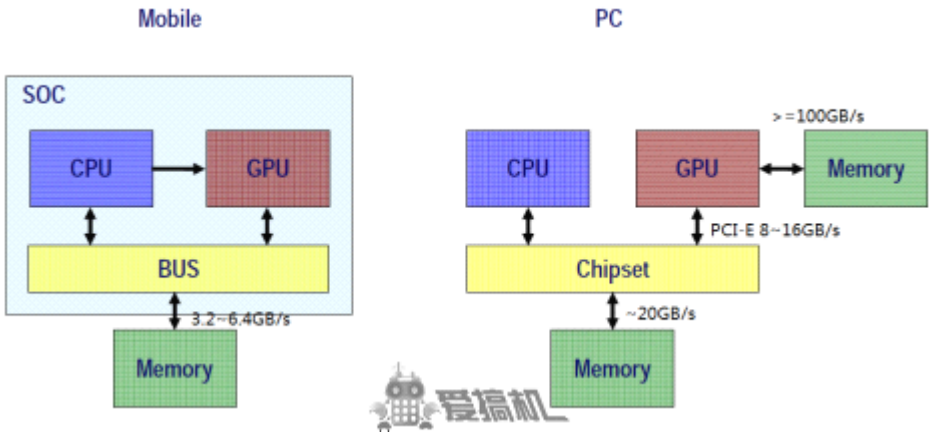
Indexed Drawing Example
Vertices [A,B,C,D] occupy positions 0, 1, 2, and 3 in the vertex list, so drawing index list [0,1,2,1,3,2] produces the geometry shown, drawing two triangles while shading only four vertices.

同样，填充率反映了 GPU 的像素输出能力。但是厂家给出的理论值，很多都是 没有贴图，没有 Shader 计算，仅仅是生成无色点的能力，跟实际使用的情况有较大的差距。又比如 Imagination Technologies 的 PowerVR SGX 系列给出的填充率，并不是实际值，而是实际值乘以了一个2.5x 的系数。这是由于 PVR GPU 架构的特殊性，可以剔除画面中被遮蔽的部分，不作渲染，减少了无用功。200MHz 的 SGX540，原生的填充率是400M，而由于这种技术的存 在，IMG 认为其等效填充率相当于1000M。在实际的场景中，如果遮蔽的部分较多，这个系数可能远超过2.5x。当然，如果场景中遮蔽较少，这个系数相 应的也会变小。

事实上三角形生成率和像素填充率作为衡量 GPU 性能的参数，在 PC 平台上，几年前就已经被淘汰了。由于从 DX8开始，现代 GPU 都已经由可编程的 Shader 来代替固定功能的单元实现各种特效，所以 Shader 的计算能力成为很重要的一点，在移动平台也是一样的。

移动平台的特点和移动 GPU 的架构

不过呢，移动平台相比于 PC 平台，还是有很多不同的。从本质上看，是由功耗和体积两方面限制的，对于图形处理来说，主要是两点：



第一，是有限的带宽。实际上，要增加计算能力，在功耗允许的情况下，堆核心 并不是一

件难事，事实上我们也看到了不少 SOC 集成了四核乃至“16核”GPU。但是难点在于，需要有足够的带宽去满足这颗强大的 GPU，避免其出现“饿 死”的情况。在左图的移动平台中，CPU、GPU 和总线被共同集成在一颗芯片上，称之为 SOC。整个 SOC，包括其中的 CPU 和 GPU，共享有限的内存带 宽。即使是相对高端的，采用64bit 内存位宽的一些 SOC，如三星4412，高通8064等，也只是6.4 - 8.5GB/s 的带宽，相比起 PC 平台主内存十几 GB/s 的带宽，和 PC GPU GDDR5显存动辄几十，不少都超过100GB/s 的带宽，只能说是少的可怜。相对另类的苹果在 iPad4中，给 A6X 芯片搭配了128bit 的 LPDDR2-1066，带宽达到了17GB/s，用以喂饱强大的 SGX 554 MP4 GPU，但相比 PC 平台依旧是小巫见大巫。因此，移动平台要在有限的带宽下实现合理的性能，在不少时候，瓶颈可能并不在于计算能力上，而在于带宽上。

第二，相比 PC 平台的 CPU，移动平台的 CPU 浮点较弱，在 Cortex-A9开始虽然有所好转，但 64bit 的 NEON 跟桌面128bit 甚至256bit 的 SIMD 还是有显著差距，外加主频的差别。因此更多的计算也依赖硬件 Vertex Shader 去完成。

因此，移动平台的 GPU 相对于 PC 平台，也会有一些不同。我们回过头来看一下移动平台的 GPU 的一些架构。
首先是传统的 IMR（Immediate Mode Rendering）架构

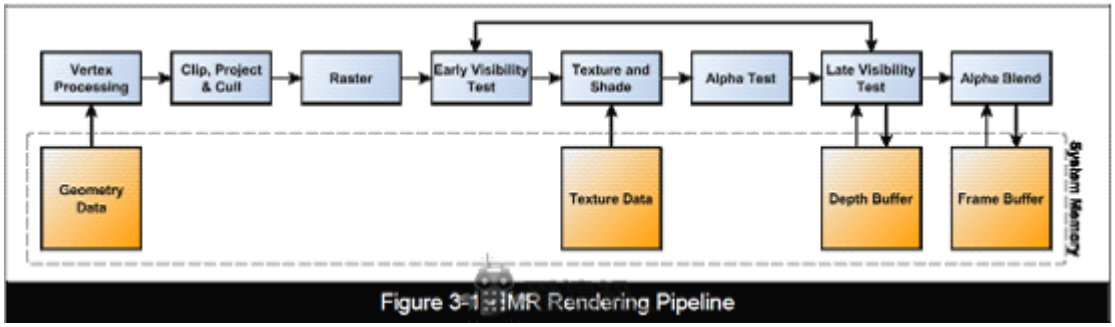
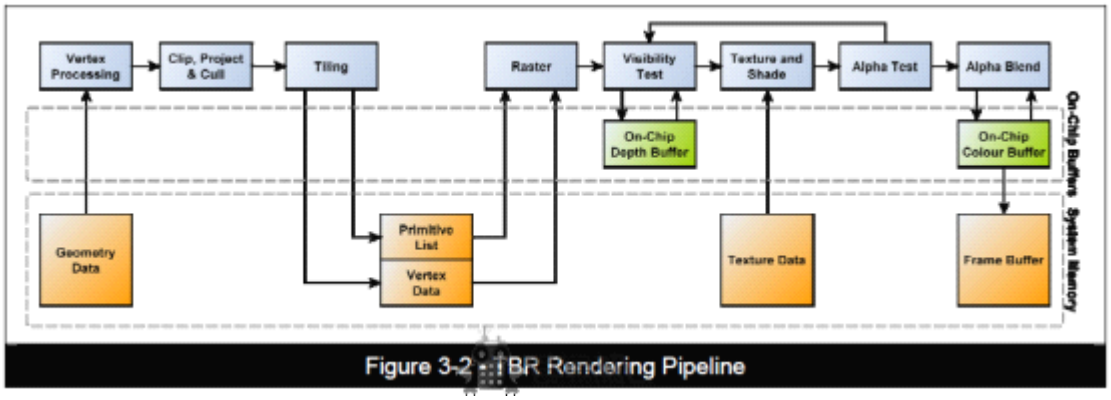


Figure 3-1 IMR Rendering Pipeline

目前几乎所有的桌面 GPU（nVIDIA，AMD）都是 IMR 架构，在移动 领域，nVIDIA 的 GeForce ULP 和 Vivante 的 GC 系列 GPU 都是属于 IMR 架构。IMR 架构的 GPU 渲染完物体后，都会把结果写到系统内存中的帧缓存里，因此就可能出现 GPU 花了大量的时间渲染了一个被遮挡的看不见的物体，而最后这些结果在渲染完遮挡物后被覆盖，做了无用功。这个问题称之为 Overdraw。虽然现代的 IMR 架构 GPU 在一定程度上可以避免这个问题，但要求应用程序将场景里的三角形按照严格的从前往后的顺序提交给 GPU，要完全避免 Overdraw 还是 很困难的。
另一方面，由于 IMR 架构的 GPU 频繁的读写和修改帧缓存，因此对带宽的要求比较高，同时也增加了电力的消耗。

所以，大部分的移动 GPU 都采用 TBR（Tile Based Rendering）的架构



ARM 的 Mali GPU 和高通的 Adreno GPU 采用 TBR（分块渲染）架构，实际上 IMG 的 PowerVR 也是分块渲染的。TBR 架构在把三角形场景变成像素图（光栅化）前，先把整个画面分成小块，这些小块的渲染在 GPU 上的高速缓存里进行，这样就避免了对帧缓存（位于系统内存里）的频繁读写和修改。当然，由于一个三角形可能被分在几个不同的块 里，三角形的数据（几何数据）可能被需要多次读取，但总的来说还是能大大减少对系统内存的访问，节约了带宽的同时也减少了电力消耗。

当然，不同的 GPU 分块的大小也有所不同，PowerVR 和 Mali 一般是16*16像素的块大小，而大部分的高通 Adreno 都带有256K 的缓存，以256K 作为块的大小进行渲染，高通称之为 binning。

但是，除去 PowerVR 外的 TBR GPU 同 IMR 一样，还是不能避免 Overdraw 的问题。

而 PowerVR 的不同之处在于，它采用的 TBDR（Tile Based Deferred Rendering）架构，可以彻底避免 Overdraw 的问题。

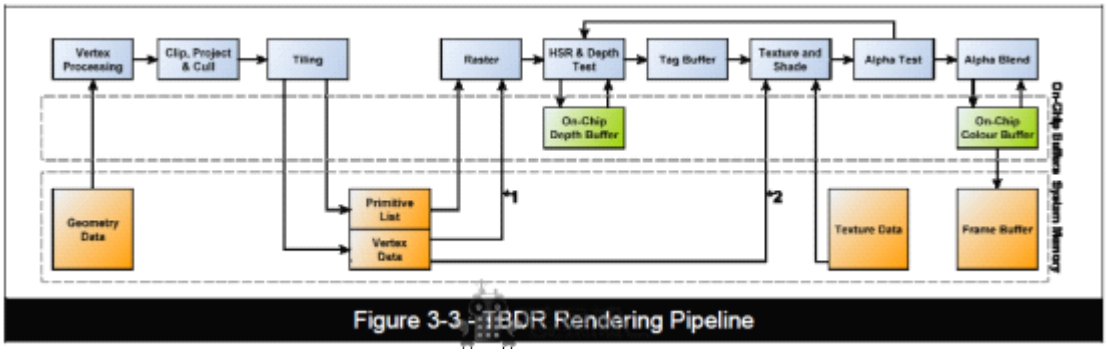
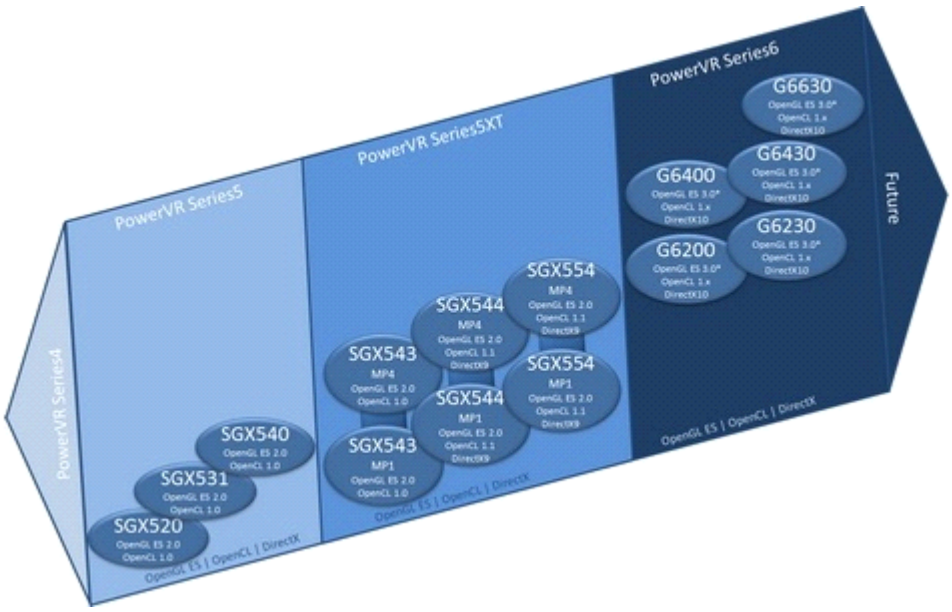


Figure 3-3 - TBDR Rendering Pipeline

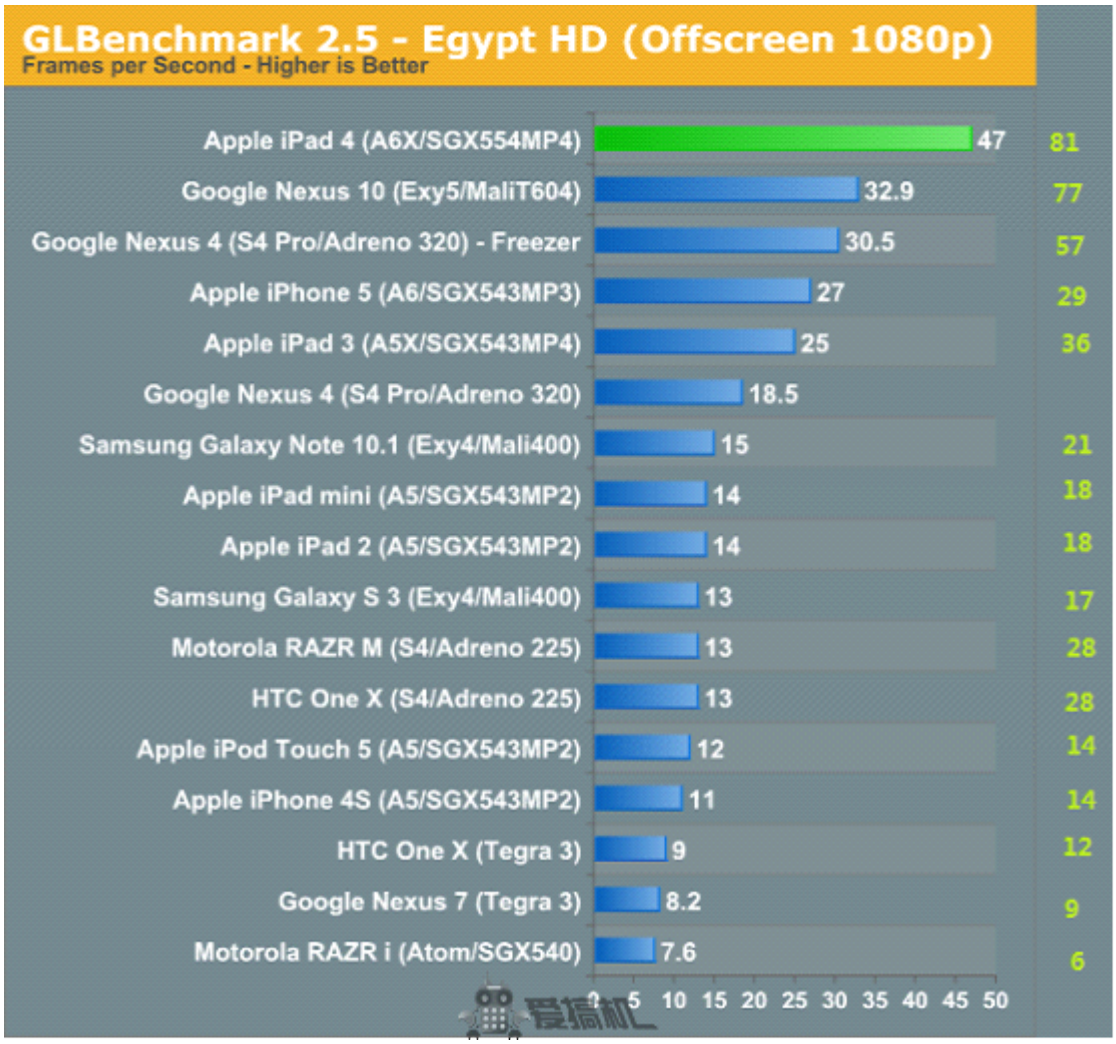
相比 TBR 更进一步的是，TBDR 在光栅化之后，有一个 HSR（Hidden Surface Removal））硬件单元，通过对一个块内的三角形进行测试，剔除掉被遮挡的三角形，合成一幅由所有可见部分组成的画面，交给后续的流水线去渲染。这样 不可见部分就不需要 Pixel Shader 去做相应的计算，也不需要去拾取相应的纹理，节省了计算量的同时也节省了带宽，对移动设备来说有很大的帮助。



在上一篇[移动 GPU 解读]中，对移动 GPU 的架构、相关参数进行了介绍，本部分介绍的则是移动 GPU 的 Shader、GPU 兼容性、“多核”的真相以及跑分问题。

说说被忽略的 Shader

接 下来我们回到 Shader。Shader 是 GPU 里负责计算的主要部分，同时占得面积最大，耗电也最多。当今的桌面 GPU 往往都不再谈三角形生成率，或是 像素填充率了，给的指标都是 Shader 的计算能力——GFLOPS。可见，Shader 性能会越来越重要。移动 GPU 也有着这样的趋势。我们看一下 Anandtech 测试的各款 GPU 的 GLBenchmark 的 Egypt HD 1080p Offscreen 得分：



右边的绿色数字是该 GPU 在 FP16精度下大致的计算性能，单位为 GFLOPS。可以看到除去个别的 GPU，Egypt 成绩跟 Shader 计算能力的相关性还是比较明显的。

先做一点铺垫：

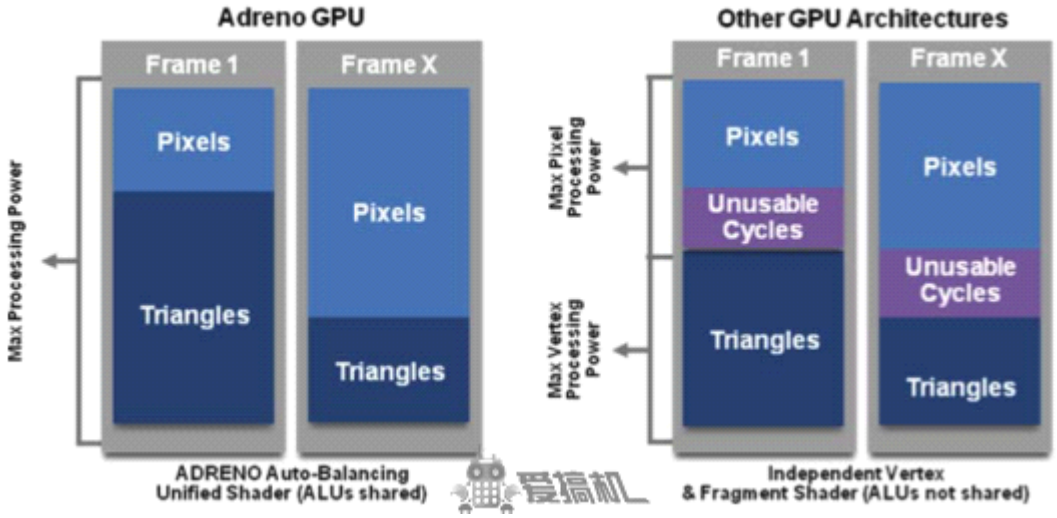
首先，**对 于浮点数做一次加法或者乘法，都算是一次操作，记作1 FLOPS。浮点数是有一定的精度的，比如16bit 的浮点数，精度就是 FP16。再高一点的 FP32就是32bit 的浮点数，也就是常说的单精度；更高 的就是64bit 的双精度 FP64了。一般来说，只有 FP32和 FP64的操作才能算是 FLOPS。**

在移动平台的 OpenGL ES 里，可以指定高、中、低三种不同的精度。对不同的 GPU 来说，高、中、低精度的实际值略有差别。如下图所示：

GPU	type	Vertex FLOAT			Fragment FLOAT		
		highp	mediump	lowp	highp	mediump	lowp
PowerVR SGX540 OMAP4460	unified	fp32 (s23e8)	fp16 (s10e5)	fix10	fp32 (s23e8)	fp16 (s10e5)	fix10
PowerVR SGX535 S5PC100	unified	fp32 (s23e8)	fp16 (s10e5)	fix10	fp32 (s23e8)	fp16 (s10e5)	fix10
PowerVR SGX543MP2 A5	unified	fp32 (s23e8)	fp16 (s10e5)	fix10	fp32 (s23e8)	**fp16? (s10e5)?**	
ULP GeForce(8) Tegra2	discrete	fp32 (s23e8)	**fp16 (s10e5)**	**fix10?**	-	**fp16 (s10e5)**	fix10
Mali-400MP4 Exynos4210	discrete	fp32 (s23e8)	fp32 or **fp16? (s10e5?)**		-	fp16 (s10e5)	
Adreno 200/205/220 系	unified	**fp32 (s23e8)**			**fp32 (s23e8)**		
GC860 JZ4770	unified	fp32 (s23e8)			fp32 (s23e8)		

对于 Adreno 和 GC 系列，**无论何种选择何种精度，都会按照 FP32精度进行计算**。而 Mali-400 和 Tegra 的 ULP GeForce 的 Pixel Shader 部分不支持高精度，**最高只支持中等的 FP16精度**。绝大部分游戏的 Pixel Shader 计算都采用中等（FP16）的精度，而 Vertex Shader 的计算一般是 FP32的精度。

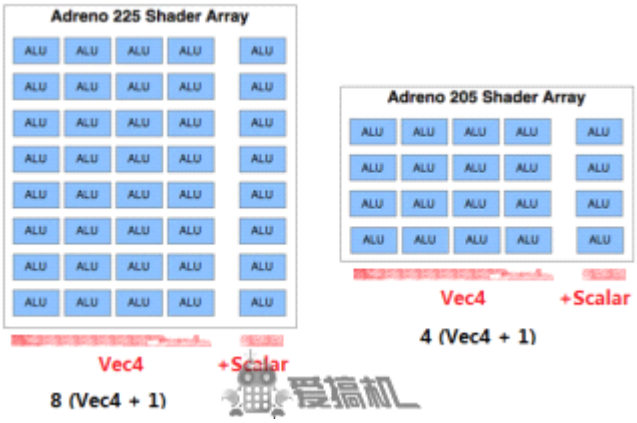
其 次，关于统一渲染架构（Unified Shader）和分离的渲染架构（Discrete Shader）。前者的 Shader 既能进行 Vertex 的计算，也能进行 Pixel 的计算，例如 PowerVR，Adreno，GC 系列。后者的 Vertex Shader 和 Pixel Shader 是分开的，典型的比如 Mali-400和 ULP GeForce。相对来说，统一渲染架构的 Shader 利用率会高些，在遇到三角形特别多像素特别少，或者相反的情况下，Shader 的计算能力不容易被 浪费。



最后，由于顶点坐标（xyzw）和像素颜色（rgba）都具有四个属性，为 了提高效率，Shader 往往被设计成 Vec4的 SIMD，也就是可以对四个数据进行打包，然后用一条指令同样的处理。当然如果数据少于四个，计算能力就 被浪费了。也有设计成一次只能处理一个数据的标量（scalar）单元。

各家 GPU 的 Shader 组成

1. 高通 Adreno 系列



Adreno 系列为统一渲染架构，shader ALU 为典型的 Vec4 + Scalar，Vec4每周期可以处理4个 FP32的 MAD 运算（乘加运算，记为2 FLOPS），Scalar 单元不能做 MAD，所以，
一个 Adreno 的 Shader 单元，每周期可提供的浮点操作数为4×2 + 1= 9 FLOPS。
主流 Adreno GPU 运算能力：
Adreno 200， 2 Vec4+1， 133MHz，2.4GFLOPS
Adreno 205， 4 Vec4+1， 266MHz，9.5GFLOPS
Adreno 220， 8 Vec4+1， 266MHz，19.1GFLOPS
Adreno 225， 8 Vec4+1， 400MHz，28.8GFLOPS
Adreno 320， 如果是16Vec4+1， 跑400MHz 的话，就是57GFLOPS
以上都是 FP32的计算能力，由于 OPENGL ES 里，高中低精度 Adreno 都是按照 FP32跑，因此在低精度的情况下，也不能获得性能提升。

2. PowerVR SGX 系列
2.1 旧的 SGX5系列
包 括 SGX530/531/535/540/545，其 Shader 计算单元为 USSE。USSE 一个周期，可以对4个 FX10（10bit 的定点数，比 FP16精度更低） 或者 2个 FP16 或者 1个 FP32 进行 MADD 操作。由于 FP32才算是正常意义上的 FLOPS，所以其性能每周期2 FLOPS。但是当两个 FP32的操作共享一个操作数时，USSE 也可以在一个周期里处理，此时就是2个 FP32的 MAD 操作，4 FLOPS。所以，USSE 的 FP32性能，每周期为2～4 FLOPS。
主流 SGX5 GPU 运算能力：
SGX530， 2USSE，200MHz，0.8～1.6 GFLOPS
MTK 的 SGX531，2USSE，300MHz，2USSE，1.2～2.4 GFLOPS
三星蜂鸟 SGX540，4USSE，200MHz，1.6～3.2 GFLOPS
OMAP4460，ATOM Z2460的 SGX540 400MHz， 4USSE，3.2～6.4 GFLOPS
不过在 FP16下，也就是大多数游戏的 Pixel Shader 精度下，相比 FP32的最差情况下就能翻倍了。同样在更低精度的 FX10下，还能再翻倍。

2.2 SGX 5XT 系列
包 括 SGX543/544/554，和它们的各种多核版本。其 Shader 计算单元为 USSE2。USSE2不像之前那样了，是个 Vec4+scalar 的 架构，单周期支持4个 FP32的 MAD 操作，外加一个简单的 scalar 操作（ADD/MUL），这样跟 Adreno 一样，每周期 9 FLOPS。
单个543/544包含4个 USSE2，性能基本一样，544多一些 DX 的 API 支持。单个554则包含8个

USSE2。
主流 SGX5XT GPU 运算能力：
iPhone 4S 里的543MP2，2×4=8个 USSE2，200MHz，14.4 GFLOPS
OMAP4470里的单个544，384MHz，4USSE2，跟上面类似
全志 A31里的544MP2，所谓的8管线就是8USSE2，300MHz，也有21.6 GFLOPS
iPad3里的 A5X，543MP4，16USSE2，250MHz，36 GFLOPS
iPad4的 A6X，554MP4，32USSE2，280MHz，就突破80 GFLOPS 了

在运算较低精度的 FP16时，USSE2的性能还能有一定的提升。

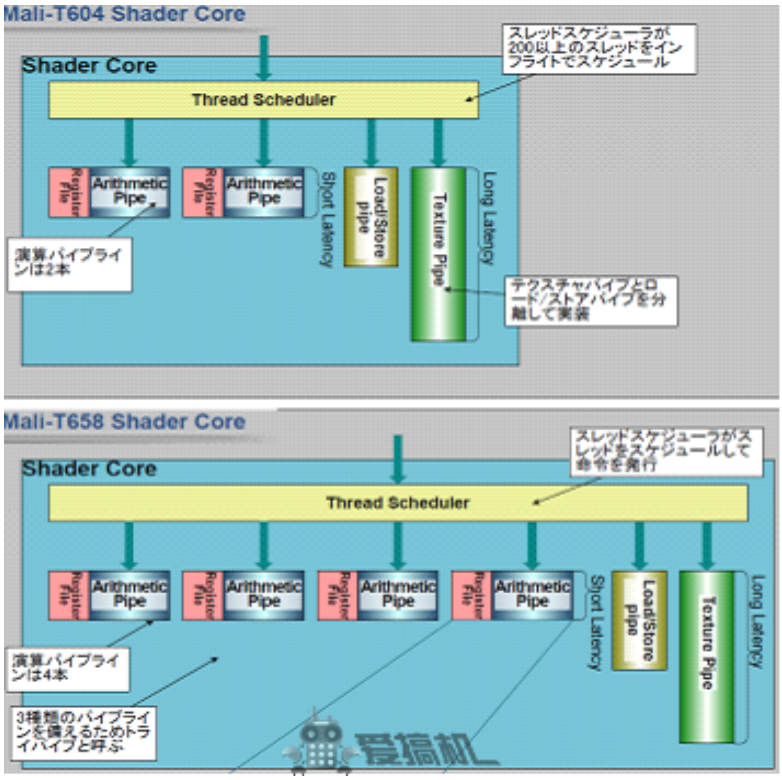
3. ARM Mali 系列

3.1 Mali-400

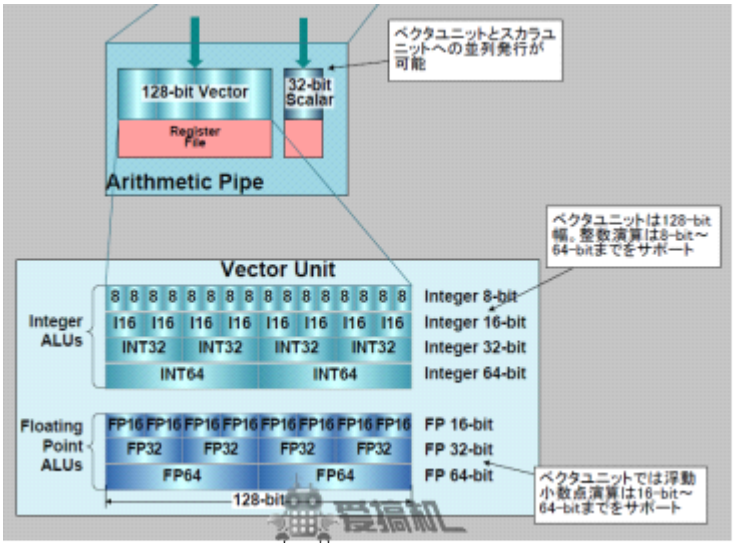
Mali-400并非 Unified Shader，是顶点和像素处理分开的
一个顶点处理器包含一个 Vertex Shader ，Vec4，支持 FP32精度
一个像素处理器包含一个 Vec4的 Pixel Shader，以及一个 TMU， Shader 支持 FP16精度
主流 Mali GPU 运算能力：
一个 Mali-400“单核”，400MHz 下，计算能力为6.4 GFLOPS
Exynos 4210的 Mali-400 MP4，266MHz，则为10.6 GFLOPS
Galaxy S3的 Mali-400 MP4， 440MHz，则为17.6 GFLOPS
Note2的 Mali-400 MP4，运行在533MHz，则为21 GFLOPS
当然这些都是 FP16…… 因为 Mali-400的 Pixel Shader 不支持 FP32精度。

3.2 Mali-T6xx 系列

T6xx 采用新架构，Shader 为统一渲染架构。其中 T604/624/628 ，一个核心含有2个 ALU，而 T658/678，为强化计算型，一个 core 含有4个 ALU。



每个 ALU 是个128bit wide 的 Vector Unit 加上一个32bit 的 Scalar 单元组成。



所以，单精度（FP32）性能为每周期9个，同 USSE2。
那么 Exynos 5250里533MHz 的 Mali-T604 四核，FP32的计算能力为 38.4 GFLOPS
同样，因为游戏里用的多的 Pixel Shader 是 FP16精度，而 T604的 VU ALU 此时处理能力能翻倍变成8个，这样每周期就是8*2 + 1 =17个。符合 ARM 宣称的500MHz 下单个 T604核心 17GFLOPS，四核心68GFLOPS 的数据。
那么 Exynos 5250里533MHz 的 Mali-T604 四核，FP16的计算能力为 72.5 GFLOPS

4. GeForce ULP
GeForce ULP 同 Mali-400，是分离的 Shader 架构。其 Vertex Shader 和 Pixel Shader 都是 Scalar 的，并非 Vec4。顶点支持 FP32精度，像素部分支持 FP20和 FX10精度。所以，
“8核”Tegra 2，4VS + 4PS，300MHz,计算能力为4.8 GFLOPS
“12核”Terga3，4VS + 8PS，520MHz，计算能力为12.5 GFLOPS

5. Vivante 的 GC 系列
跟 Adreno 差不多，也是 Vec4 +1的结构，同样高中低精度都按照 FP32计算，低精度下不会有提升。
RK29的 GC800， 1Vec4+1， 450MHz，4 GFLOPS
飞思卡尔 i.MX6的 GC2000， 4Vec4+1，600MHz，21.6 GFLOPS
海思 K3V2的 GC4000，8Vec4+1，480MHz，34.6 GFLOPS

GPU“兼容性”
现在还有个经常被提到的是 GPU 的“兼容性”问题，这里就要涉及到各个 GPU 支持的纹理格式了。
首先是 ETC1，这个是 OPENGL ES 2.0支持的纹理格式，大家都得支持。但这个纹理的一个缺点是不支持 alpha 通道，所以对于有 alpha 通道的纹理，就要拆成2个纹理去读取，效率低，浪费了带宽。
而 PVRTC 是 PowerVR 自家的纹理格式，同样 ATITC 是高通 Adreno 的纹理格式，此外 S3TC

就是桌面很常见的 DXT，微软 DirectX 3D 的纹理格式，这些都是支持 alpha 通道的。
PowerVR GPU 支持自家的 PVRTC 和通用的 ETC1（iOS 下的 PVR GPU 只支持 PVRTC），Adreno 支持自家的 ATITC 和通用的 ETC1,NV 的 GeForce 和 Vivante 的 GC 系列支持 DXT 和 ETC1，剩下 Mali-400只支持 ETC1。所以，对应不同的 GPU，会有不同的游戏数据包。通用数据包，一般都会采用 ETC1，虽然通用，但由于不支持 alpha 通道要贴图2次，对于非 Mali 的 GPU 其实都算是吃亏了。如果用自己支持的其他格式，就不用受这个苦了。对于贴图单元 （TMU, Texture Mapping Unit）数目相对较少的 Adreno 2xx 系列，恐怕更是吃亏。
当然，纹理的支持度只是兼容性的一方面，并不是兼容性问题的全部。

GPU	圧縮Tex
PowerVR SGX 535	PVRTC
PowerVR SGX 543MP2	PVRTC
PowerVR SGX 540	PVRTC/ETC1
Z430	ATITC/3DC/ETC1
Adreno 200	ATITC/3DC/ETC1
Adreno 205	ATITC/3DC/ETC1
Adreno 220	ATITC/3DC/ETC1
Tegra250 ULP GeFroce(8)	S3TC/LATC/ETC1
Tegra3 ULP GeFroce(12)	S3TC/LATC/ETC1
ZMS-08HD	S3TC/ETC1
Mali-400MP4	ETC1
GC860	S3TC/ETC1

各家的“多核”
GPU 硬件的部分基本说完了，这里总结一个表格，同时给出了 GPU 厂商官方定义的一个“核”的内容，谁的核里料多，谁比较不厚道，应该也是一目了然了吧。面对各种“16核”“8管线”的宣传，大家也应该能比较清楚的辨别了吧。

GPU	Shader	TMU	官方定义的一个“核”
PVR SGX530/531	2 USSE	1	-
PVR SGX535	2 USSE	2	-
PVR SGX540	4 USSE	2	
PVR SGX543/544	4 USSE2(Vec4+1)	2	4 USSE2+2TMU+其他（完整核心）
PVR SGX544 MP2	8 USSE2(Vec4+1)	4	4 USSE2+2TMU+其他（完整核心）
PVR SGX543 MP4	16 USSE2(Vec4+1)	8	4 USSE2+2TMU+其他（完整核心）
PVR SGX554	8 USSE2(Vec4+1)	2	8 USSE2+2TMU+其他（完整核心）
PVR SGX554 MP4	32 USSE2(Vec4+1)	8	8 USSE2+2TMU+其他（完整核心）
Adreno 200	2 Vec4+1	1	-
Adreno 205	4 Vec4+1	1	-
Adreno 220/225	8 Vec4+1	2	-
Adreno 320	16 Vec4+1?	4?	-
Mali-400	1GP(Vec4)+1PP(Vec4)	1	-
Mali-400 MP4**“四核”**	1GP(Vec4)+4PP(Vec4)	4	PP(Vec4 Pixel Shader + 1TMU)
Mali-T604/T624“四核”	8 Vec4+1	4	(2 Vec4 + 1TMU)
Mali-T628 “八核”	16 Vec4+1	8	(2 Vec4 + 1TMU)
Mali-T658/T678 “八核”	32 Vec4+1	8	(4 Vec4 + 1TMU)
ULP GeForce (Tegra 2)**“八核”**	4VS(Scalar)+4PS(Scalar)	1	1Vertex Shader 或者 1Pixel Shader
ULP GeForce (Tegra 3) **“12 核”**	4VS(Scalar)+8PS(Scalar)	2?	1Vertex Shader 或者 1Pixel Shader
Vivante GC800 **“单核”**	1 Vec4+1	1	1 Vec4 Shader
Vivante GC1000 **“双核”**	2 Vec4+1	1	1 Vec4 Shader
Vivante GC2000 **“四核”**	4 Vec4+1	2	1 Vec4 Shader
Vivante GC4000 **“八核”**	8 Vec4+1	4	1 Vec4 Shader

跑分跟实际表现不一样？优化很重要！

最后，规格只是 GPU 的一个方面，实际表现跟架构也有很大的关系。更进一步的，就算是 Benchmark 中跑分差不多的 GPU，在不同的游戏中，实际表现也会有差别。

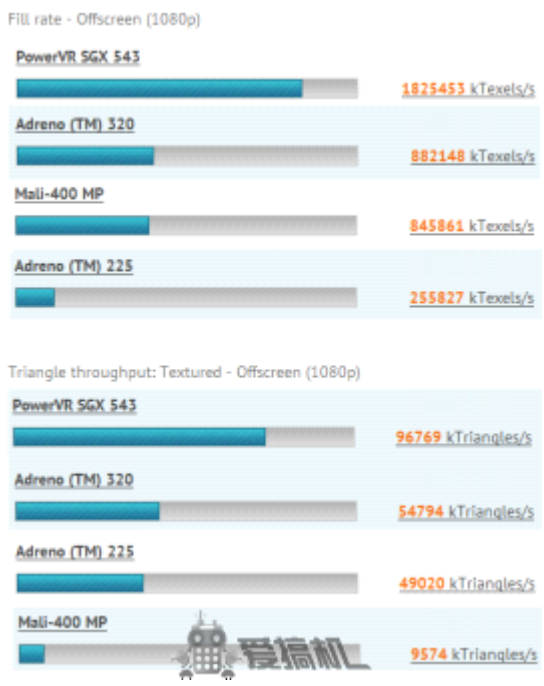
首先，**Benchmark 程序，大部分都是公平的，所以本质上，Benchmark 都是“零优化”程序，公平起见，他们的纹理会用 RGBA 的 PNG，TGA，或者 ETC1纹理，不会用到各个 GPU 自家的格式。**

但是游戏不一样，游戏可以做相应的优化。例如 PVR 的 GPU，可以用4bpp 甚至2bpp 的 PVRTC

纹理，相比于未压缩的贴图就可以节省8倍甚至16倍的带宽。而没有被优化到的情况下，可能只 能跟着 Mali 用不支持 alpha 通道的 ETC1，做2次贴图，浪费带宽。部分厂商甚至在通用数据包里放了一些未压缩贴图，那差距就更大了。同款游戏，跑 分接近的 GPU，iOS 上的特效更好，流畅度更佳，就有优化的原因。

其次，Benchmark 在一定程度上都是相对超前的。大部分 GPU 跑 Benchmark 的帧率，都不会到流畅的级别（要是满帧了还怎么测出区别）。早期的 Benchmark 可能更加侧重贴图和像素部分。**新一代的 Benchmark 则提升了场景复杂度，对多边形和 Shader 计算的压力进一步增大，例如 GLBenchmark 从2.1到2.5的提升。**因此，一些三角形生成能力和原生 Shader 计算能力高的 GPU，比如 Adreno 220/225，得分提升就会比较明显。而 Mali-400则在2.5中遇到三角形生成的瓶颈，得分表现不如之前。

而游戏是给人玩的，**终端厂商或是 SOC 厂商可以跟游戏厂商合作，针对 GPU 的特点进行相应的优化。** 不同 GPU 侧重很不一样，比如 Mali-400，三角形很弱，像素部分，填充率强。高通 Adreno 2xx，Vivante 的 GPU，多边形很强，Shader 计算强，但填充率较弱。如果场景对 Mali 优化，游戏商可以减少画面中多边形的量，用贴图和像 素部分实现更多的特效。这样对 Adreno 2xx 系列不利。如果对 Adreno 优化，则可以增加场景复杂度，用更多的三角形进行更精细的建模，但这样对 Mali 则不利。这只是一方面，在一些细节 上，还可以进行更深层次的优化，各家的 GPU 也都会提供相应的工具。



最后，GPU 的跑分在一定程度上能反映 GPU 的实际性能，但最终在游戏中的表现还是很看厂商优化的。**所以也不要一味的盯着跑分，多问问玩过的朋友，多看看实测，会更有帮助。**