3.9 合并阶段

如第2.5.2节所述，合并阶段是将单个片元（在像素着色器中生成）的深度和颜色与帧缓冲区组合的阶段。DirectX将此阶段称为输出合并；OpenGL将其称为逐样本操作。在大多数传统的流水线图（包括我们自己的）上，这个阶段是模板缓冲区和z缓冲区操作发生的地方。如果片元可见，则在此阶段发生的另一个操作是颜色混合。对于不透明的表面，不涉及真正的混合，因为片元的颜色只是替换了之前存储的颜色。片元和存储颜色的实际混合通常用于透明度和合成操作（第5.5节）。

想象一下，由光栅化生成的片元通过像素着色器运行，然后在应用z缓冲区时发现被某些先前渲染的片段隐藏。所有在像素着色器中完成的处理都是不必要的。为了避免这种浪费，许多GPU在执行像素着色器之前执行一些合并测试[530]。片元的z深度（以及任何其他正在使用的东西，例如模板缓冲区或裁剪）用于测试可见性。如果隐藏，则片元被剔除。此功能称为early-z[1220, 1542]。像素着色器能够改变片段的z深度或完全丢弃片段。如果发现像素着色器程序中存在上述任一类型的操作，则early-z通常无法使用并被关闭，通常会降低管道效率。DirectX 11和OpenGL 4.2允许像素着色器强制进行early-z测试，但有许多限制[530]。有关early-z和其他z-buffer优化的更多信息，请参阅第23.7节。有效地使用early-z会对性能产生很大的影响，这将在第18.4.5节中详细讨论。

合并阶段占据固定功能阶段（如三角形设置）和完全可编程着色器阶段之间的中间地带。虽然它不是可编程的，但它的操作是高度可配置的。特别是可以设置颜色混合来执行大量不同的操作。最常见的是涉及颜色和alpha值的乘法、加法和减法的组合，但其他运算也是可能的，例如最小值和最大值，以及按位逻辑运算。DirectX 10添加了将来自像素着色器的两种颜色与帧缓冲区颜色混合的功能。此功能称为双源颜色混合，不能与多个渲染目标结合使用。MRT不支持混合，而 DirectX 10.1引入了对每个单独的缓冲区执行不同混合操作的功能。

正如上一节末尾所提到的，DirectX 11.3提供了一种通过ROV使混合可编程的方法，但以性能为代价。ROV和合并阶段都保证绘制顺序，也就是输出不变性。无论生成像素着色器结果的顺序如何，API要求将结果按输入顺序、逐个对象、逐个三角形进行排序并发送到合并阶段。