5.7.2　功能单元的性能

图 5-12 提供了 Intel Core i7 Haswell 参考机的一些算术运算的性能，有的是测量出来的，有的是引用 Intel 的文献[49]。这些时间对于其他处理器来说也是具有代表性的。每个运算都是由以下这些数值来刻画的：一个是延迟(latency)，它表示完成运算所需要的总时间；另一个是发射时间(issue time)，它表示两个连续的同类型的运算之间需要的最小时钟周期数；还有一个是容量(capacity)，它表示能够执行该运算的功能单元的数量。

运算	整数			浮点数		
	延迟	发射	容量	延迟	发射	容量
加法	1	1	4	3	1	1
乘法	3	1	1	5	1	2
除法	3 ~ 30	3 ~ 30	1	3 ~ 15	3 ~ 15	1

图 5-12　参考机的操作的延迟、发射时间和容量特性。延迟表明执行实际运算所需要的时钟周期总数，而发射时间表明两次运算之间间隔的最小周期数。容量表明同时能发射多少个这样的操作。除法需要的时间依赖于数据值

我们看到，从整数运算到浮点运算，延迟是增加的。还可以看到加法和乘法运算的发射时间都为 1，意思是说在每个时钟周期，处理器都可以开始一条新的这样的运算。这种很短的发射时间是通过使用流水线实现的。流水线化的功能单元实现为一系列的阶段(stage)，每个阶段完成一部分的运算。例如，一个典型的浮点加法器包含三个阶段(所以有三个周期的延迟)：一个阶段处理指数值，一个阶段将小数相加，而另一个阶段对结果进行舍入。算术运算可以连续地通过各个阶段，而不用等待一个操作完成后再开始下一个。只有当要执行的运算是连续的、逻辑上独立的时候，才能利用这种功能。发射时间为 1 的功能单元被称为完全流水线化的(fully pipelined)：每个时钟周期可以开始一个新的运算。出现容量大于 1 的运算是由于有多个功能单元，就如前面所述的参考机一样。

我们还看到，除法器(用于整数和浮点除法，还用来计算浮点平方根)不是完全流水线化的——它的发射时间等于它的延迟。这就意味着在开始一条新运算之前，除法器必须完成整个除法。我们还看到，对于除法的延迟和发射时间是以范围的形式给出的，因为某些被除数和除数的组合比其他的组合需要更多的步骤。除法的长延迟和长发射时间使之成为了一个相对开销很大的运算。

表达发射时间的一种更常见的方法是指明这个功能单元的最大吞吐量，定义为发射时间的倒数。一个完全流水线化的功能单元有最大的吞吐量，每个时钟周期一个运算，而发射时间较大的功能单元的最大吞吐量比较小。具有多个功能单元可以进一步提高吞吐量。对一个容量为 C，发射时间为 I 的操作来说，处理器可能获得的吞吐量为每时钟周期 C/I 个操作。比如，我们的参考机可以每个时钟周期执行两个浮点乘法运算。我们将看到如何利用这种能力来提高程序的性能。

电路设计者可以创建具有各种性能特性的功能单元。创建一个延迟短或使用流水线的单元需要较多的硬件，特别是对于像乘法和浮点操作这样比较复杂的功能。因为微处理器芯片上，对于这些单元，只有有限的空间，所以 CPU 设计者必须小心地平衡功能单元的数量和它们各自的性能，以获得最优的整体性能。设计者们评估许多不同的基准程序，将大多数资源用于最关键的操作。如图 5-12 表明的那样，在 Core i7 Haswell 处理器的设计中，整数乘法、浮点乘法和加法被认为是重要的操作，即使为了获得低延迟和较高的流水