Design Automation for Dilution of a Fluid Using Programmable Microfluidic Device–Based Biochips

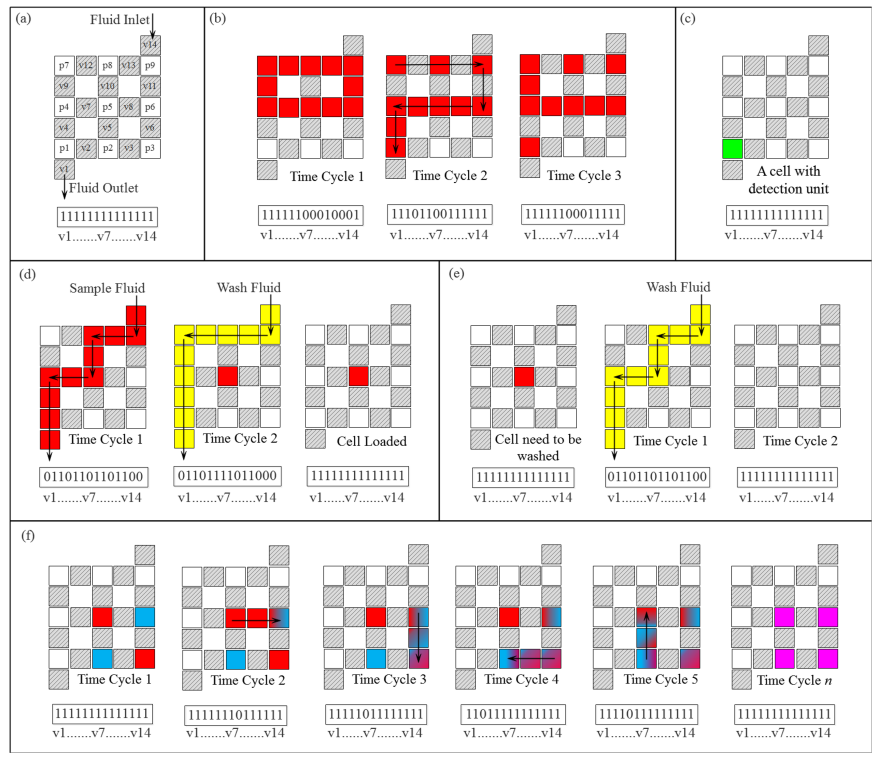


图1 FPVA芯片结构及各功能示意图

文中FPVA生物芯片的建模如图1（a）所示，p表示流通道，v表示阀门，流通道周围被阀门包围，红色的流体能够通过阀门打开如图1（b）所示停滞或在芯片上流动，而在通道单元上可以执行各种功能，如图1（c）所示的检测单元来实现流体试剂的各项检测。从图1（d）的时间周期1中可以观察到，其他几个连通的单元也被相同的样品流体加载;因此，我们需要另一种流量（即洗涤流量）来从这些连通的单元中洗涤这些样品流体。

样品制备（PMSP）的性能指标为*Tms*、*W*、*S*、*B*，其中*Tms*、*W*、*S*和*B*分别表示混合步骤数、样品、缓冲液和废物单位。芯片设计（PMCD）另一类性能指标，即*T*、*P*、*A*，对于自动化真实的片上生物协议而言，与PMSP同样重要，其中*T*、*P*和*A*分别代表时间、功耗和面积。执行时间（*T*）是流体的装载（*Tl*）、清洗（*Tw*）、混合（*Tm*）和布线（*Tr*）时间的累积和（*Tr*）（即，*T*=*Tl*+*Tw*+*Tm*+*Tr*），混合时间大于装载、洗涤和布线时间。每个加载周期之后都有一个清洗周期，以消除加载周期期间单元的意外填充。

由于求解过程将资源与先前的资源相邻，因此它不需要流体的布线，这隐含地最小化了开销*Tr*以及最小化PMCD中的*T*、*P*、*A*的目标。这里，功率*P*表示为在芯片上执行生物测定而进行的瓣膜驱动总数。显然，最小化阀门驱动可以降低功耗，并延长芯片的使用寿命。

文章提出了一种资源放置算法，即广义架构映射算法（GAMA），该算法以锯齿方式将资源放置在芯片上，以便最优地利用芯片区域。此外，流体分配算法（FAAP）将预期的

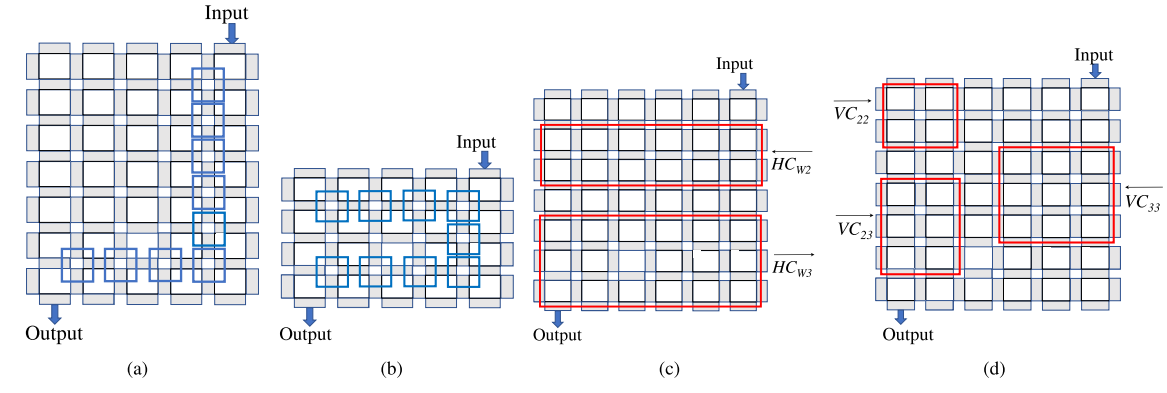


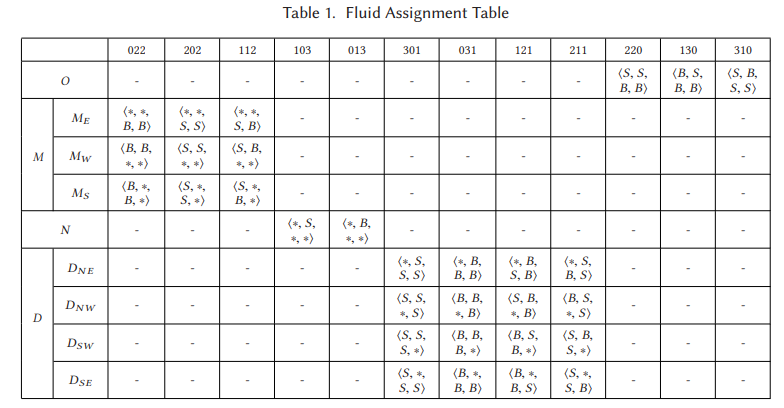
图2（a）混合器（“蓝色”中的方框）以曼哈顿方式在FPVA芯片上的放置表示。（b）混合器以锯齿状方式在FPVA芯片上放置表示。（c-d）FPVA布局中不同尺寸的水平和垂直通道分配的示意图

流体（样品、缓冲液、试剂等）加载到GAMA指定的单元格中。最后，提出了一种流体加载算法（FLAP），该算法在一个加载周期内加载同一流体的多个FPVA单元。因此，减少负载循环的总数可以减少FPVA的阀门切换并延长FPVA的使用寿命。

资源放置是一个众所周知的NP-hard问题。给定生物协议的计划DAG（有向图）映射到FPVA以执行操作。映射可以通过多种方式完成，在无布线放置的情况下，是利用可用芯片区域的最佳方式。如图8所示，可以看出，与混合器以锯齿状方式放置（如图8（b）所示）相比，任何随机或对角线放置（如图8（a）所示）都将占用更大的芯片面积。用于将稀释树的混合步骤映射到FPVA中的混合循环。

边主要介绍流体加载的FLAP算法：

FAAP算法以流体到单元的形式返回FPVA单元的流体分配图：模式P。模式P是*W*×*H*规模的FPVA芯片模型，由2D矩阵表示，其中每个元素代表一个FPVA单元。元素（*pxy*∈P）由四重<*δ*、*x*、*y*、*F*>定义。其中，*δ*是水平，*x*和*y*表示约束条件为1≤*x*≤*W*和1≤*y*≤*H*的单元格坐标，*F*表示分配给单元格*pxy*的流体类型（例如，样品、试剂、缓冲液）。构建了一个名为*Fluid\_Assignment\_Table*的查找表，用于将混合模型映射到扫描序列元素（即*O/M*/*N*/*D*），如表1所示，该表本质上是静态的，它也可以针对通用混合模型进行修改/扩展，以便进行在线流体分配。



在没有流体装载机构自动化的情况下，阀门切换是手动计算的，并且需要相当于要装载

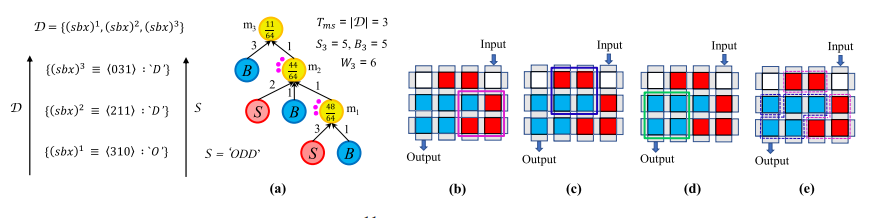


图3 （a）目标*CFCt*=11/64的稀释树及从D导出的扫描序列S。（b）放置混合器*M*1，用于在芯片上映射混合步骤*m*1（*D*1=（*sbx*）1 ≡ <310>，*S*1='*O*'），其中红色和蓝色单元分别代表样品和缓冲液，混合器以红色显示。（c） 混合器*M*2的放置（以蓝色显示），用于在片上映射混合步骤*m*2（*D*2=（*sbx*）2≡<211>，*S*2='*D*'）。（d） 混合器*M*3的放置（以绿色显示），用于在片上映射混合步骤*m*3（*D*3=（*sbx*）3≡<031>，*S*3='*D*'）。（e） 将具有相同类型流体的 PMD 单元分组，这些流体可以在一个周期内加载。加载上述所有样品需要四个周

期（流体样品需要两个周期，如洋红色所示，缓冲液需要两个周期，蓝色显示）。

的FPVA单元数量的负载清洗周期。例如，如图3所示，12个FPVA单元中有10个流体单元（5个单元格带样品，5个单元格带缓冲液）。无需自动化即可填充这些FPVA单元，需要10个负载清洗周期。流体装载的自动化可以在一个装载-清洗周期中装载相同流体的多个单元，从而减少了装载-清洗周期的数量。反过来，这也最大限度地减少了阀门驱动并延长了FPVA芯片的使用寿命。文章介绍了一种名为FLAP的算法，该算法可最大程度地减少FPVA单元中流体负载的负载清洗周期数。

FPVA架构的流体加载问题模型如下：

输入：给定流体结合模式P，芯片区域W×H。

输出：*#Load*个加载周期，*#wash*个清洗周期。

目标：最大限度地减少将流体加载到FPVA芯片中的负载清洗循环次数。

FLAP将流体与单元的结合模式P作为输入，并将需要用相同液体填充的连接单元进行分组。这些单元在一个加载循环中加载，因为这些单元位于从流体入口到流体出口的流体流动路径中。为了识别已经属于负载循环一部分的单元，单元被标记为加载单元;否则，它们将被标记为未加载的单元格。

第1步：在所有未加载的单元格中找到右下角的单元格。

第2步：找到与第1步中发现的单元相邻的最大连接单元群，该单元群充满了相同类型的流体，该单元可以在从流体入口到流体出口的路径中以曼哈顿方式找到。如果没有此类单元格，则仅选择一个单元格。

第3步：如果仍有卸载的单元格，请返回第1步。否则，执行完成。

一个说明性示例：流体到单元加载模式*P*={<1，3，1，S>，<1，3，2，B>，<1，4，1，S>，<1，4，2，S>，<2，2，2，B>，<2，2，3，S>，<2，3，2，∗>，<2，3，3，S>，<3，1，1，B>，<3，1，2，B>，<3，2，1，B>，<3，2，2，∗>}是输入。如图6（e）所示，有四个这样的基团，如红色（用于具有样品液的单元）和蓝色（用于具有缓冲液的单元）所示。因此，四个负载清洗周期（因为在这种情况下，不会产生额外的清洗开销）可以加载这些流体以产生目标*CFCt*=11/64。