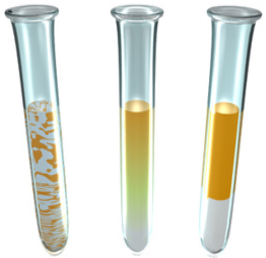
Application-Specific Fault-Tolerant Architecture Synthesis for Digital Microfluidic Biochips

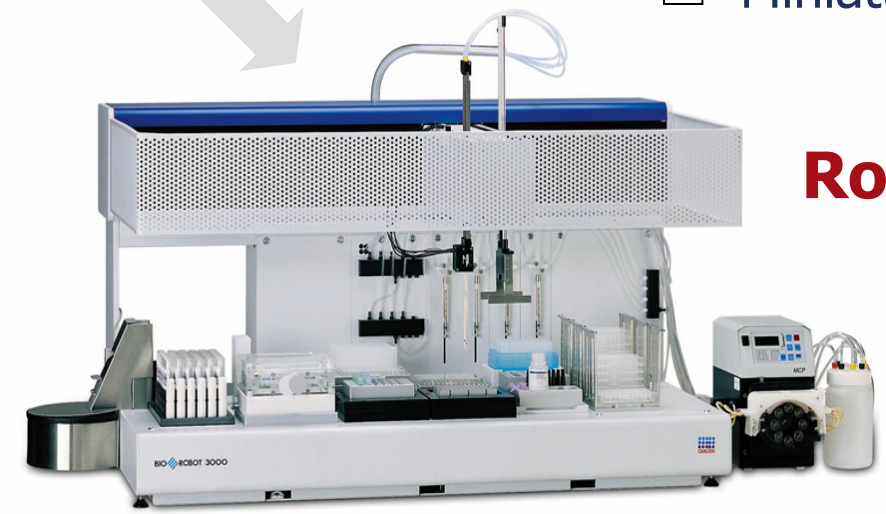
Mirela Alistar, Paul Pop, Jan Madsen
Technical University of Denmark, Lyngby

Biochips



Test tubes

- ☐ Automation
- ☐ Integration
- ☐ Miniaturization

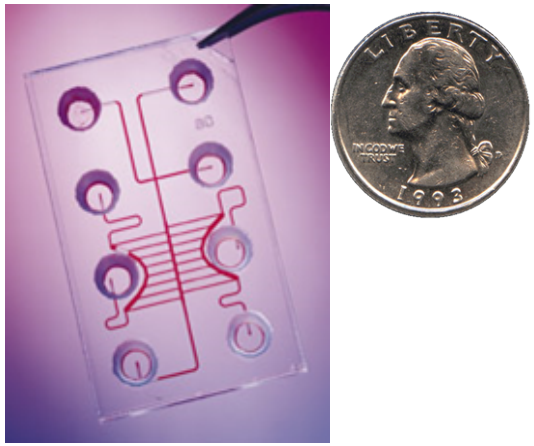


Robotics

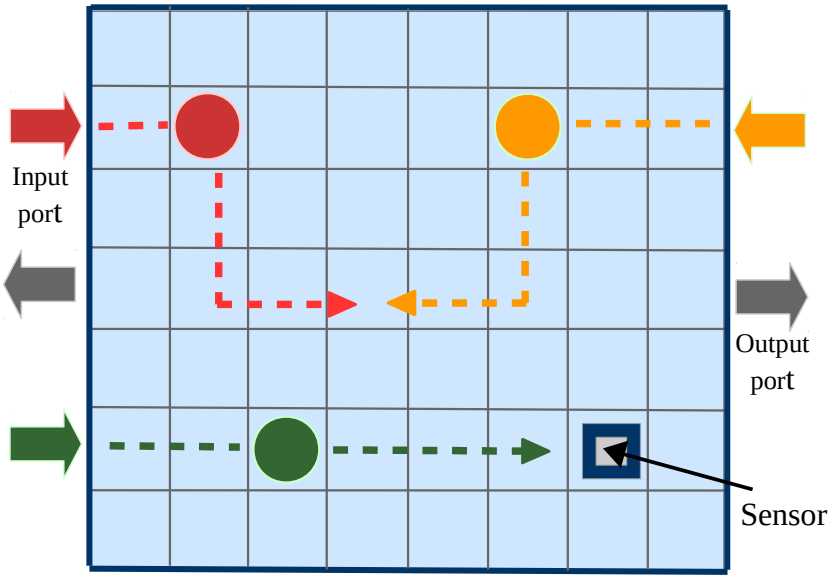
- ☑ Automation
- ☑ Integration
- ☐ Miniaturization

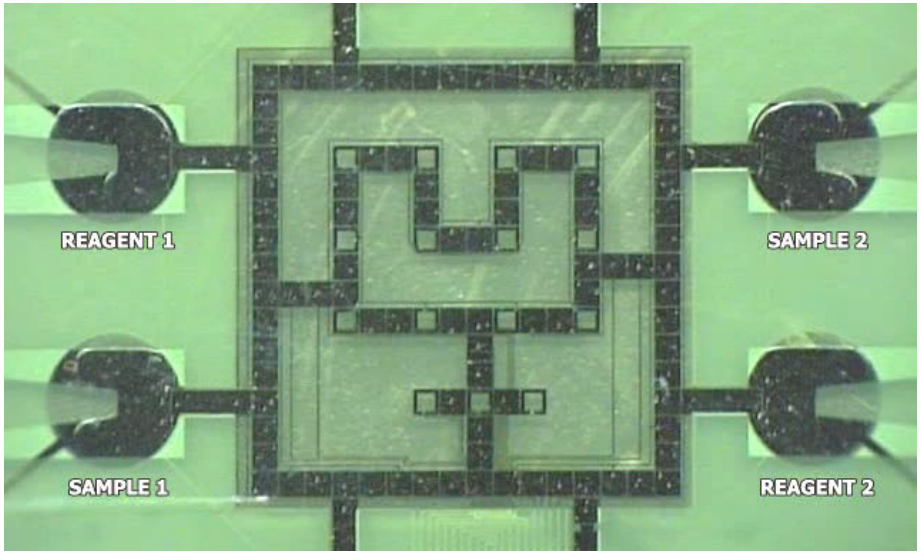
Microfluidics

- ☑ Automation
- ☑ Integration
- ☑ Miniaturization



Droplet-based Biochips

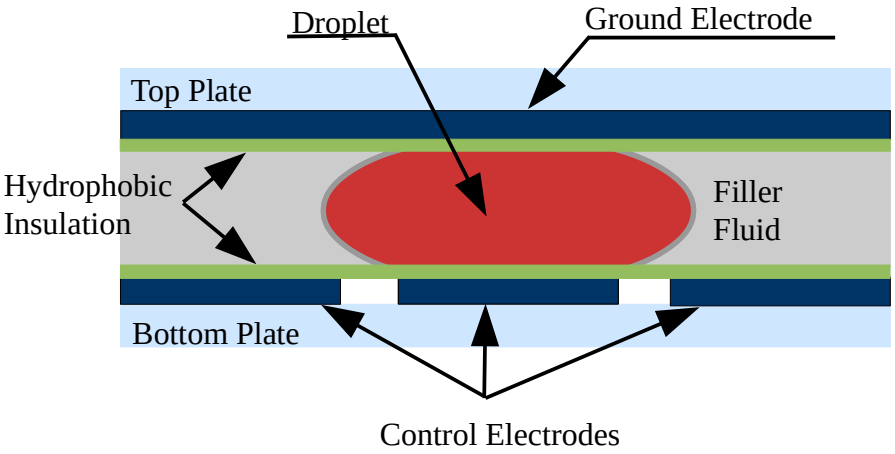




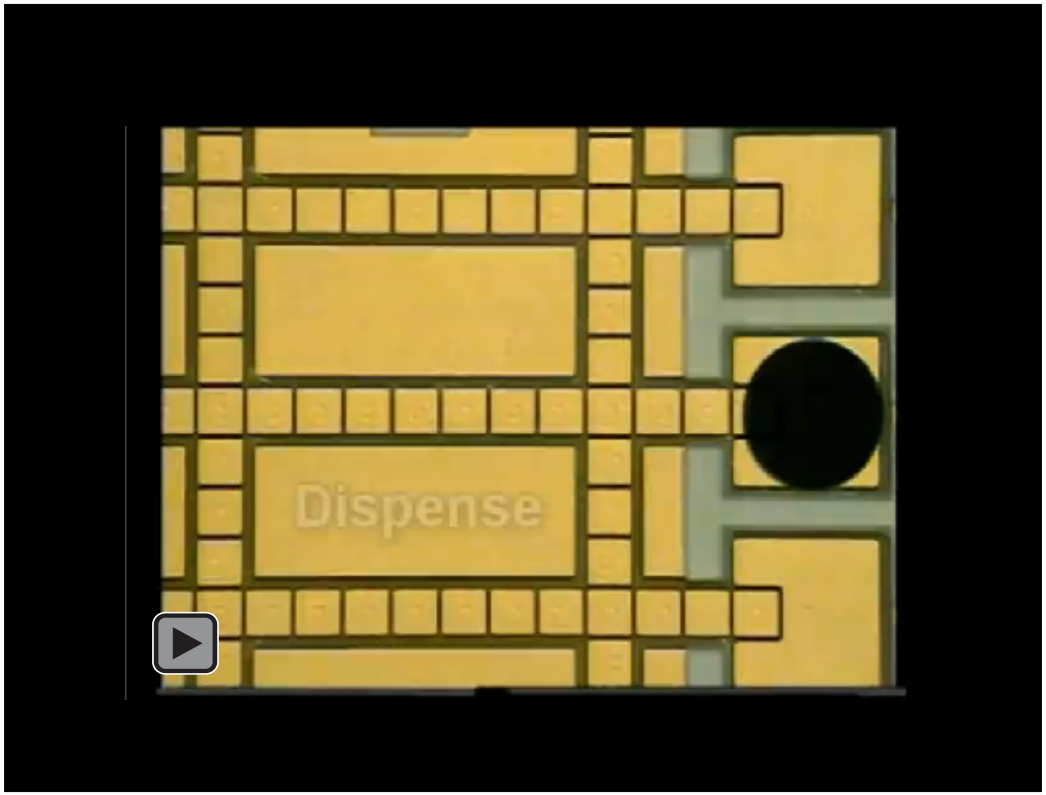
Biochip from Duke University

Digital Microfluidic Biochips (DMB)

Electrowetting on Dielectric



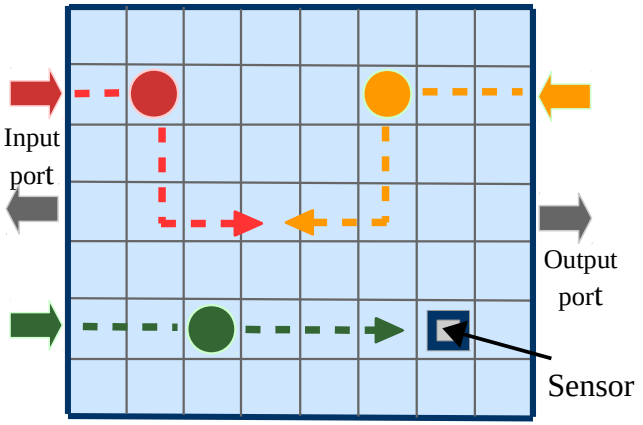
Fluidic Operations



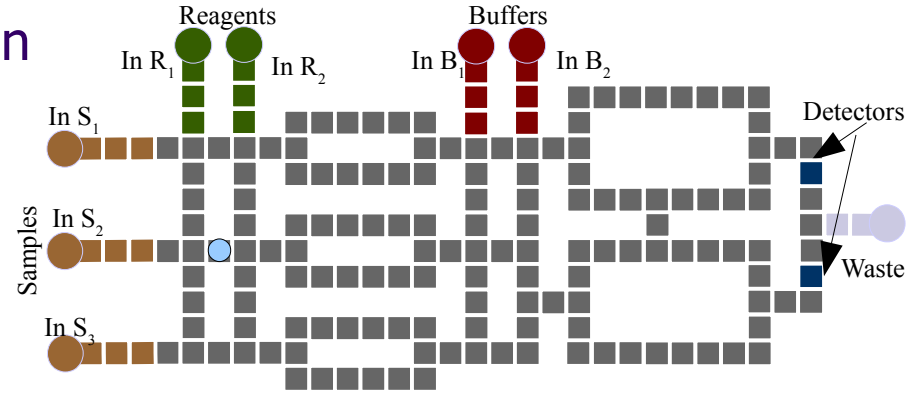
Video source: Advanced Liquid Logic http://www.liquid-logic.com/

DMB Architecture

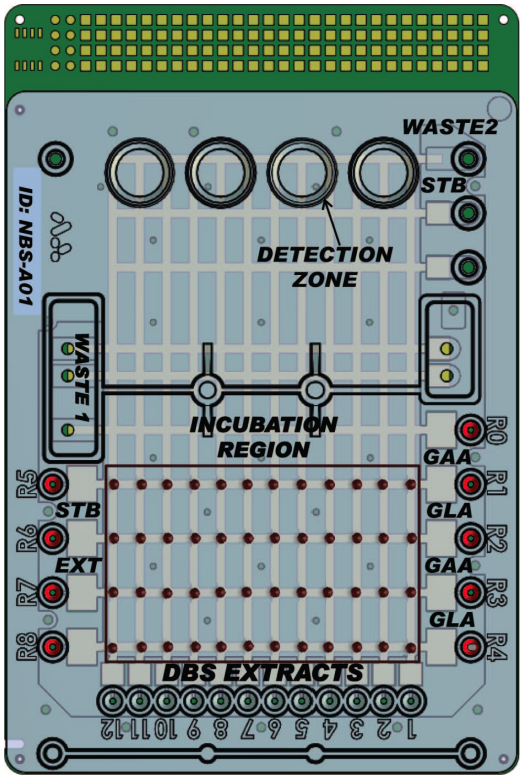
- General-Purpose Architecture
  - Reconfigurable
  - Versatile
  - Fault-tolerant



- Application-Specific Architecture
  - Designed for one application
  - Reduced costs
    - Production costs
    - Reagent costs

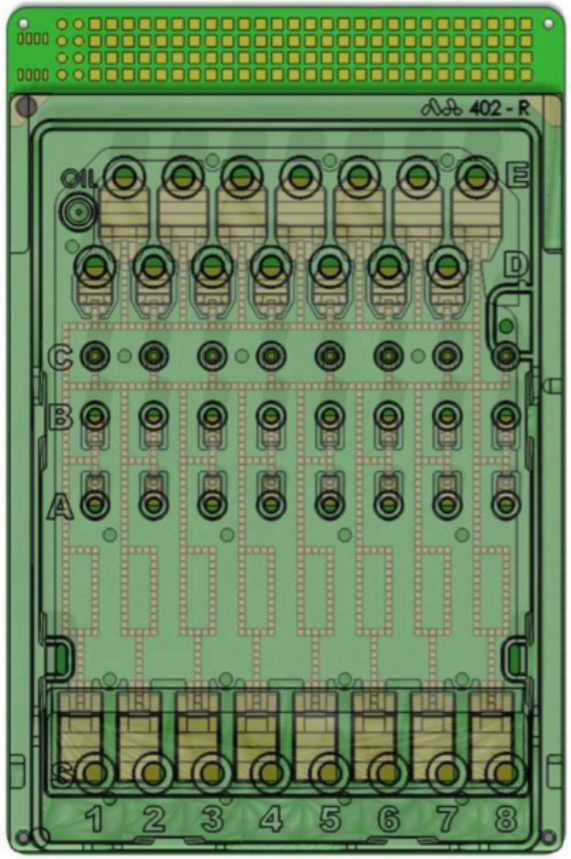


Application-Specific Biochips



Biochip for Newborn Screening

http://www.liquid-logic.com/



Biochip for Sample Preparation

http://www.nugeninc.com/

Cost Evaluation

$$Cost_{\mathcal{A}} = \sum N_{M_i} \times Cost_{M_i}$$

where

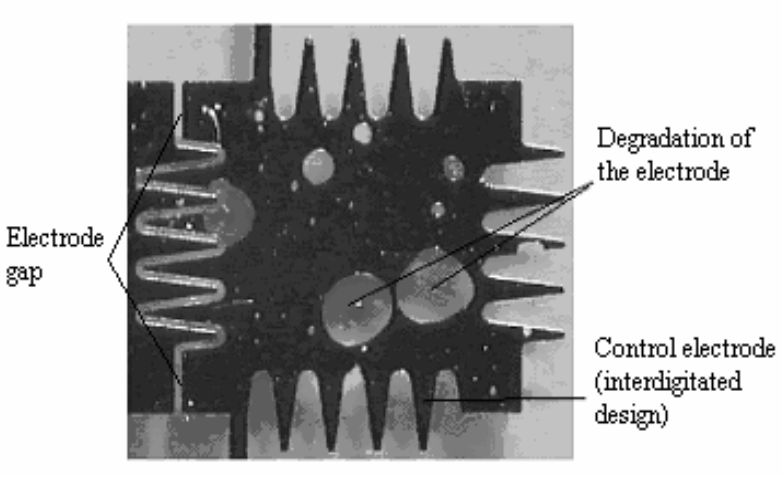
- **A** is the architecture
- $\mathbf{N_{Mi}}$ is the number of components of type M
- $\mathbf{Cost_{Mi}}$ is the cost of the physical component Mi

Component Library

Name	Unit cost	Dimensions (mm)	Time (s)
Electrode	1	1.5×1.5	N/A
Input Reservoir	3	1.5×4.5	2
Waste Reservoir	3	1.5×4.5	N/A
Capacitive Sensor	1	1.5×4.5	0
Optical Detector	9	4.5×4.5	8

Compilation: difficulties

- **Problem**: permanent faults
- **Importance**:
  - Increase the yield of DMBs
  - Improve the batch control



Insulator degradation

Image source: *Testing of Droplet-based Microelectrofluidic Systems, Fei Su 2003*

Problem: Architecture Synthesis

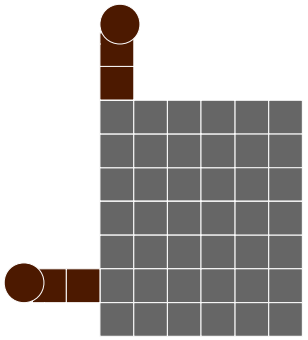
- **Given**
  - Biochemical application
  - Deadline requirements
  - Library of components (physical and virtual)
  - The number k of permanent faults

- **Determine**
  - An application-specific architecture $\mathcal{A}$, so that
    - the cost is minimized and
    - the application completes within deadline for any occurrence of the k permanent faults

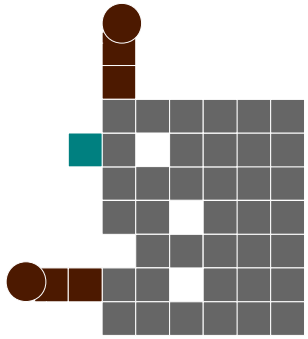
Optimization: SA moves

- Non-reconfigurable components (reservoirs, detectors)
  - Add/Remove
  - Change placement
- Reconfigurable elements (electrodes)
  - Add/Remove a single electrode
  - Add/Remove a row of electrodes on the side

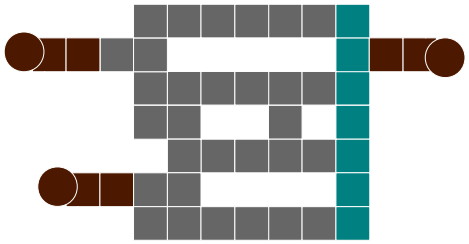
$$Objective(\mathcal{A}) = Cost_{\mathcal{A}} + W \times max(0, \delta_G^k - D_G)$$



Initial architecture

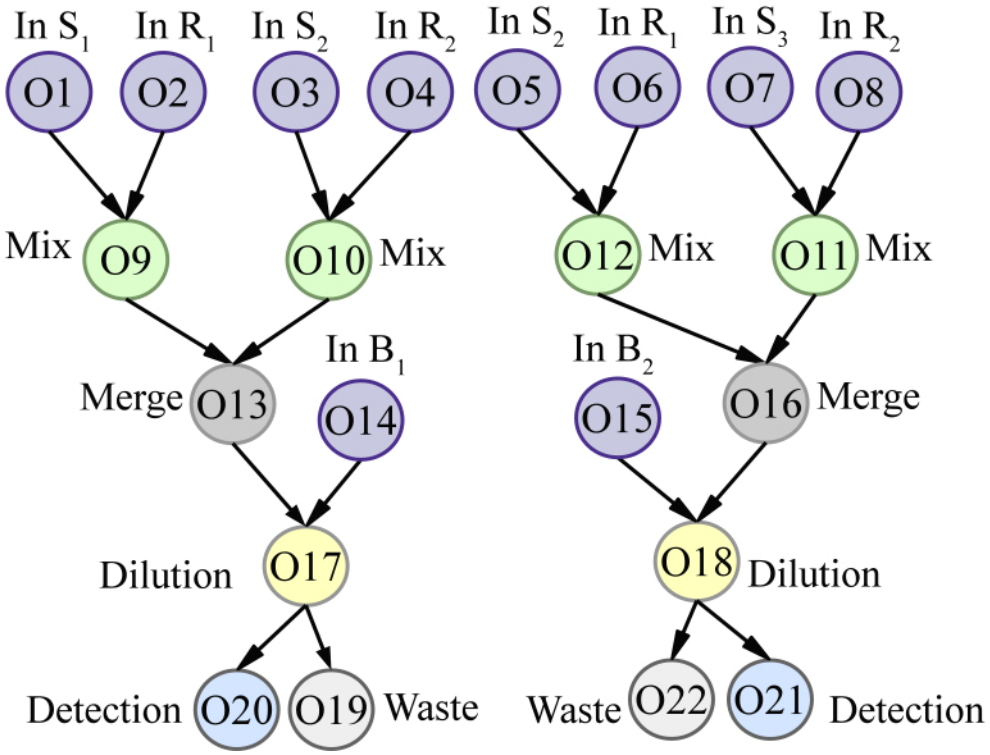


Add single electrode (green)
Remove single electrode (white)

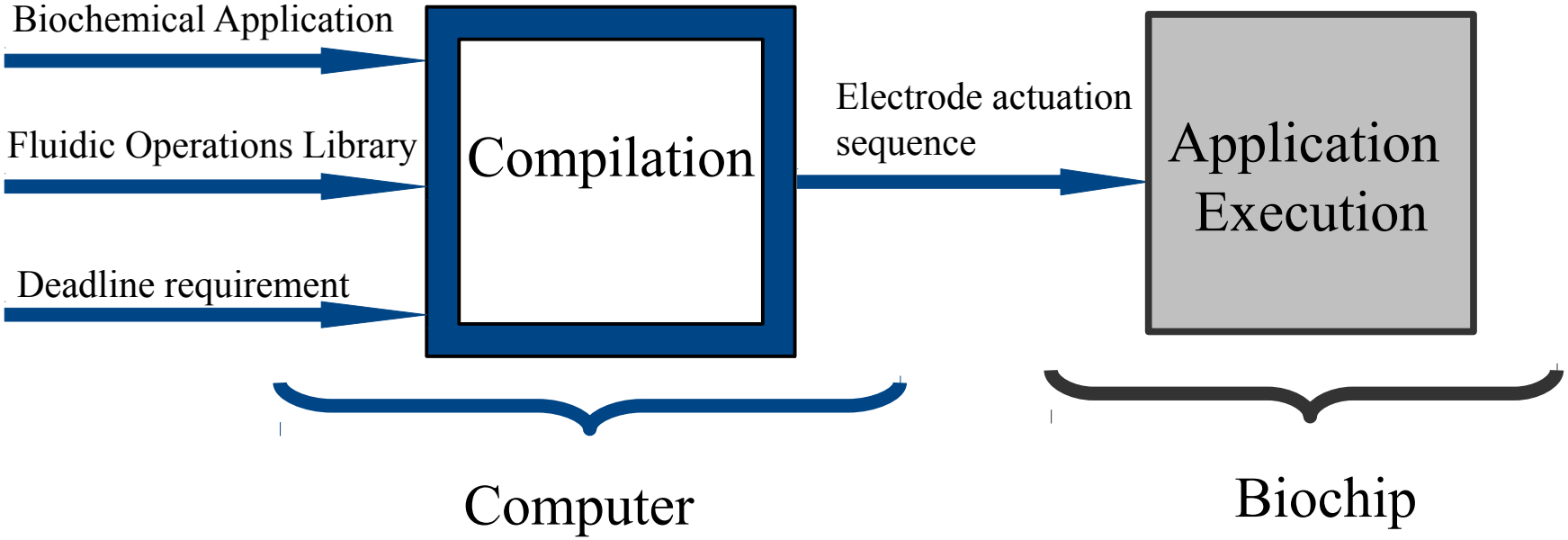


Add row of electrodes (green)
Change placement of reservoir (red)

Biochemical Application Model



Compilation Flow

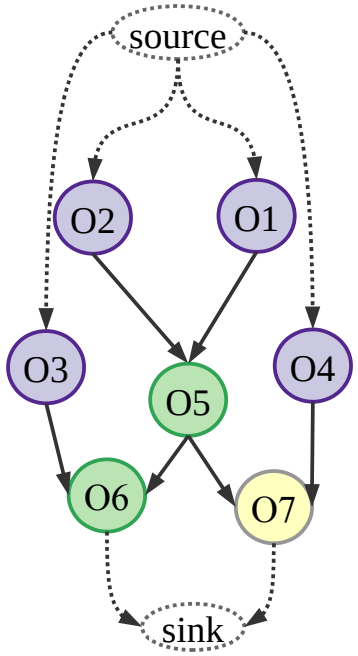


Compilation: Main steps

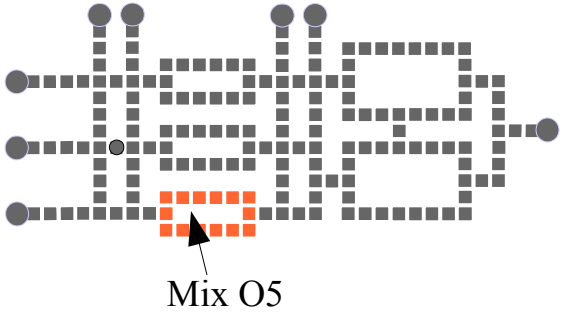
Allocation

Operation	Area	Time (s)
Mix	2x5	2
Mix	2x4	3
Mix	2x2	10
Detection	1x1	30

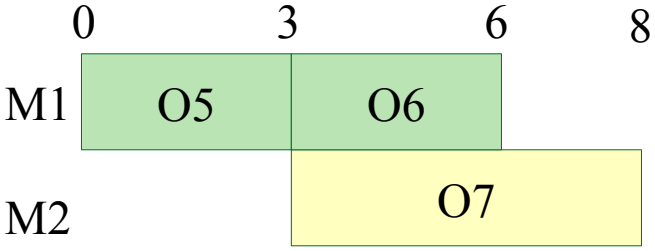
Binding



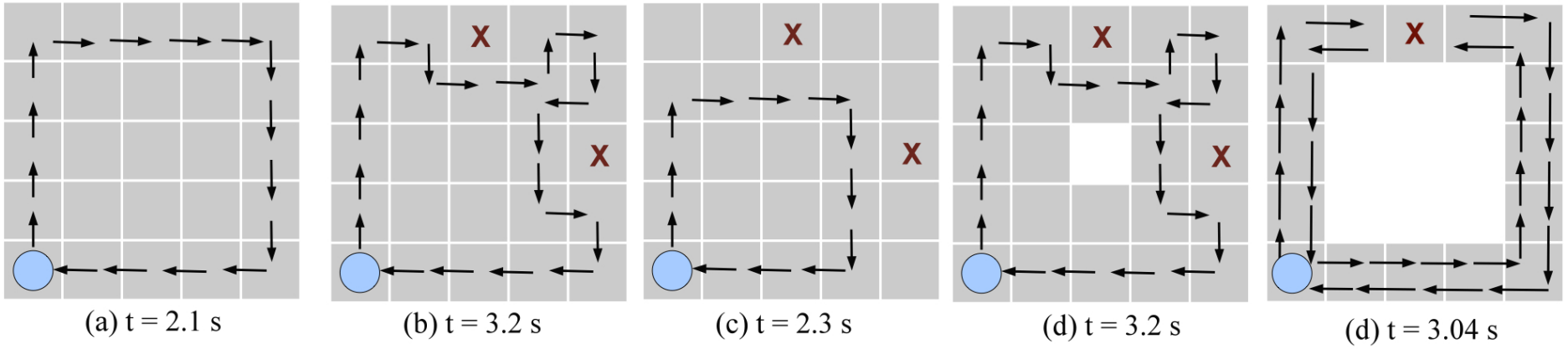
Placement



Scheduling



Fault-Tolerant Overhead



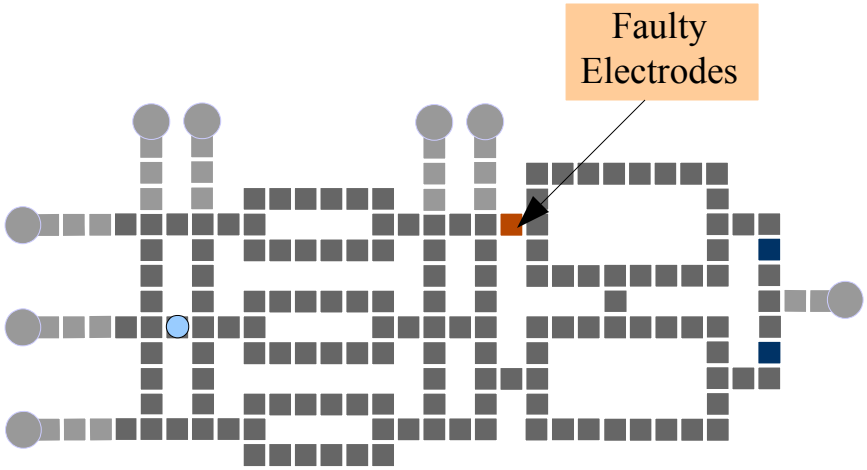
Evaluation of fault-tolerant overhead (faults are marked with X)

- Fault-tolerant overhead
  - Considers the impact of faults on the operation execution time
  - Routing-based operation execution, *Maftei 2012*

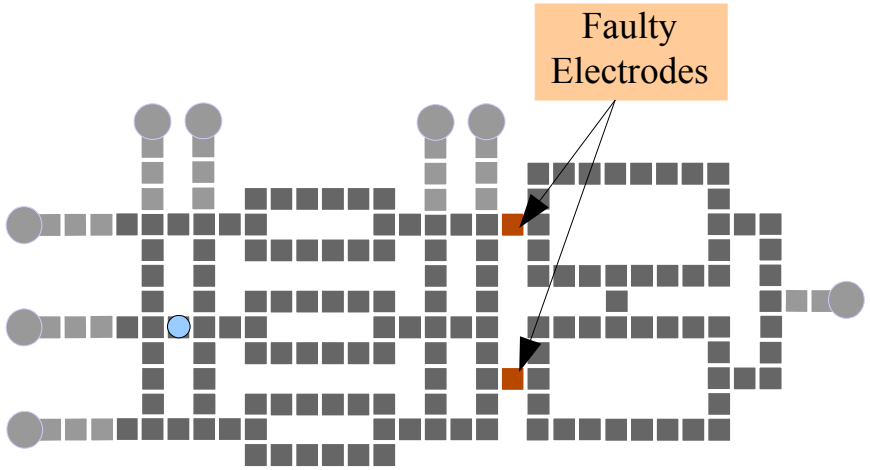
Compilation: difficulties

- **Problem**: fast compilation
- **Importance**:
  - It is part of an optimization loop
- **Solution**:
  - List-Scheduling based compilation
  - Routability test
    - Tests if, no matter where k faults are located, there is at least one route between any two electrodes

Routability test



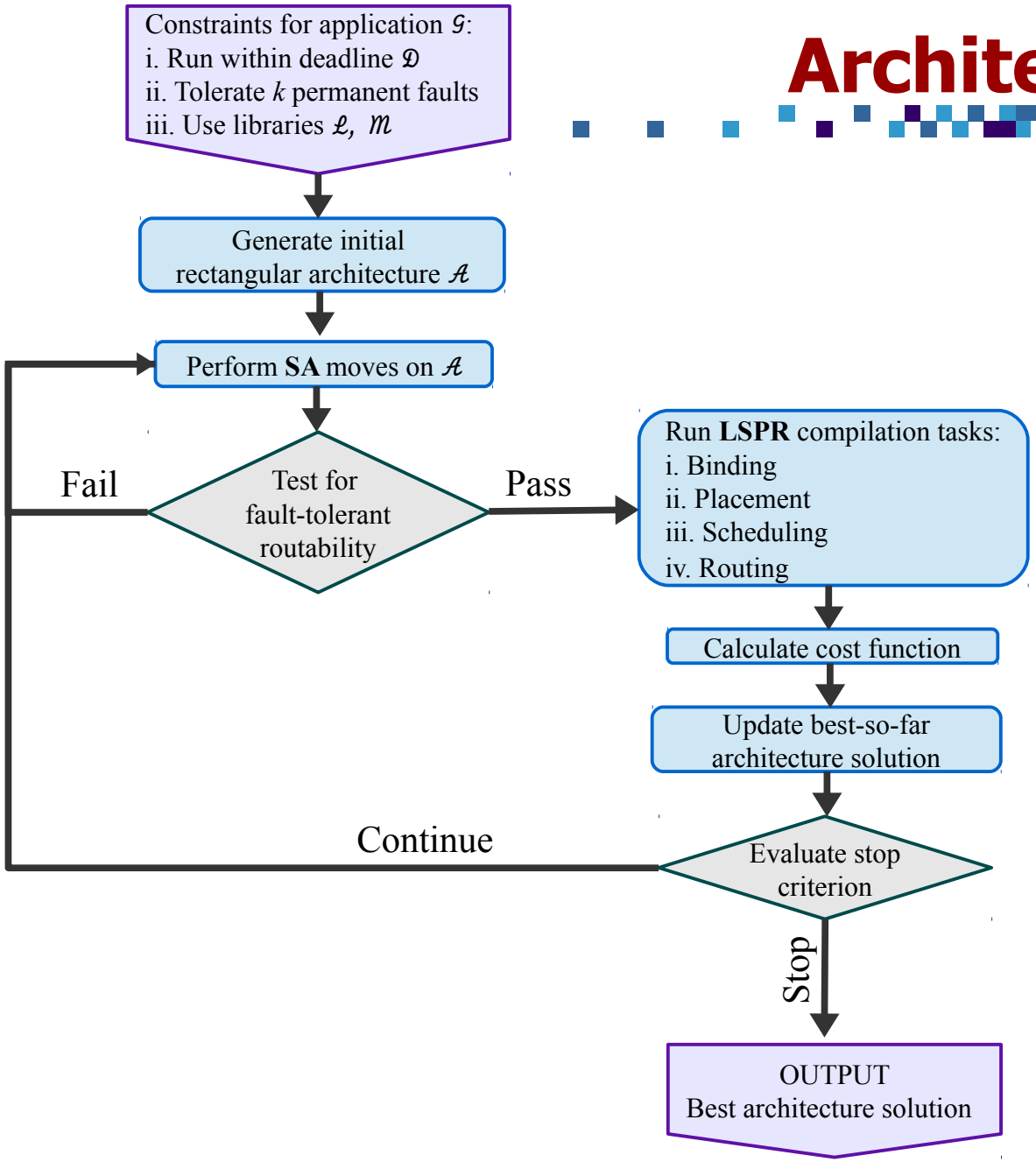
Routable architecture for 1 fault



Non-Routable architecture for 2 faults

- Routability test
  - Tests if, no matter where k faults are located, there is at least one route between any two electrodes
  - Algorithm that tests k-vertex connectivity in a graph, *S. Even (1973)*

Architecture Synthesis



- Simulated Annealing
- Test k-vertex connectivity
- List-scheduling
- Fast-template placement
- Hadlock routing

Experiments: setup

- Biochemical applications:
  - The mixing stage of polymerase chain reaction (PCR)
  - In-vitro diagnosis on human physiological fluids (IVD)
  - The colorimetric protein assay (CPA)
- Deadlines:
  - PCR – 10 s; IVD – 15 s; CPA – 100 s
- Implementation:
  - Java
- **Evaluation**:
  - Cost-effectiveness of the architectures resulted from our synthesis
  - Pessimism of List-Scheduling based compilation
  - Overhead in execution time due to permanent faults (k=0,1,2)

Experiments: Architecture synthesis

App.	$k=0$ Arch	Cost	C_{SA}	T_{SA}	$k=1$ Arch	Cost	C_{SA}	T_{SA}	$k=2$ Arch	Cost	C_{SA}	T_{SA}
PCR	7×10 (1,1,1)	79	60	14	7×10 (1,1,1)	79	65	38	9×11 (1,1,1)	108	98	50
IVD	7×10 (2,2,2)	88	62	16	7×10 (2,2,2)	88	70	58	10×8 (2,2,2)	98	85	45
CPA	7×8 (2,1,2)	71	59	10	7×8 (2,1,2)	71	66	20	11×12 (2,1,2)	147	127	30

- Applications are resulted from our synthesis (col. 4, 8, 12)
- Compared with best possible general-purpose architectures (col. 3, 7, 11)
- **Our synthesis produces cheaper architectures**

Experiments: FT Overhead

App.	Cost	$\delta^0_{\mathcal{G}}$ (s)	$\delta^1_{\mathcal{G}}$ (s)	Deviation (%)	$\delta^2_{\mathcal{G}}$ (s)	Deviation (%)
PCR	98	8.42	8.81	4.6	9.43	11.9
IVD	85	12.62	13.11	3.8	14.81	17.3
CPA	129	153.9	169.3	10	190.11	23.5

- k = 0 faults (column 2)
- k = 1 faults (column 3)
- k = 2 faults (column 6)
- Applications are resulted from our synthesis
- Comparison with the case when there are no faults
- **Average deviation is 11.8%**

Experiments: LS Compilation

App. (ops.)	Arch.	$\delta^0_{\mathcal{G}}(s)$	Exec. time	$\delta^{opt}_{\mathcal{G}}(s)$	Exec. time	Deviation (%)
PCR (7)	9×9	11	25 ms	10	60 min	9
IVD (28)	9×10	77	91 ms	73	60 min	5.4
CPA (103)	11×12	219	498 ms	214	60 min	2.3

- Near-optimal value is obtained with Tabu-Search, *Maftei 2010*
- General-purpose architectures
- No faults
- **Average deviation from near-optimal is 5.5%**

Conclusions

- SA-based architecture synthesis
  - List-Scheduling based compilation (fast)
- Reduced cost architectures
- Fault-tolerant architectures
- Increase the yield of DMBs

Backup slides

Optimization: Simulated Annealing

$\mathcal{A}^0$ - initial architecture
T^0 - initial temperature
T^L - temperature length
eps – cooling rate

$$Objective(\mathcal{A}) = Cost_{\mathcal{A}} + W \times max(0, \delta_{\mathcal{G}}^{k} - D_{\mathcal{G}})$$

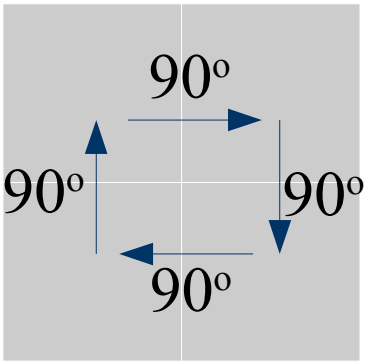
```
temp = T^0;
A = A^0;
repeat
   while (temp < T^L) do
       A^new = moves(A); //generate new architecture
       delta = Objective(A) - Objective(A^best);
       if (delta<0)
           A^best = A^new;
       elseif (Math.random < e^-delta/temp) //accept bad solutions with low probability
           A^best = A^new;
       endif
  endwhile
  temp = temp * eps;
until stop criterion is true
```

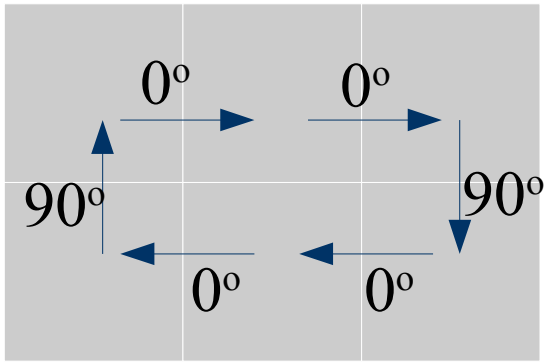
List Scheduling

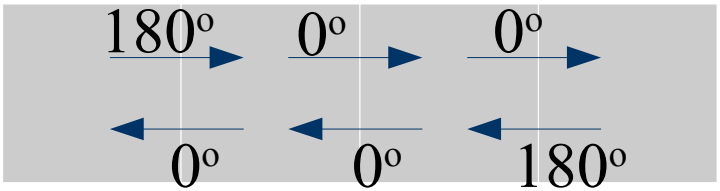
ListScheduling($Graph$, $\mathcal{C}$, $\mathcal{B}$, $\mathcal{P}$)

1 CriticalPath($Graph$)
2 **repeat**
3 $List$ = GetReadyOperations($Graph$)
4 O_i = RemoveOperation($List$)
5 t_i^{start} = Schedule(O_i, $\mathcal{B}(O_i)$, $\mathcal{C}$, $\mathcal{P}$)
6 t = earliest time when a scheduled operation terminates
7 UpdateReadyList($Graph$, t, $List$)
8 **until** $List = \emptyset$
9 **return** S

Routing-based Operation Execution







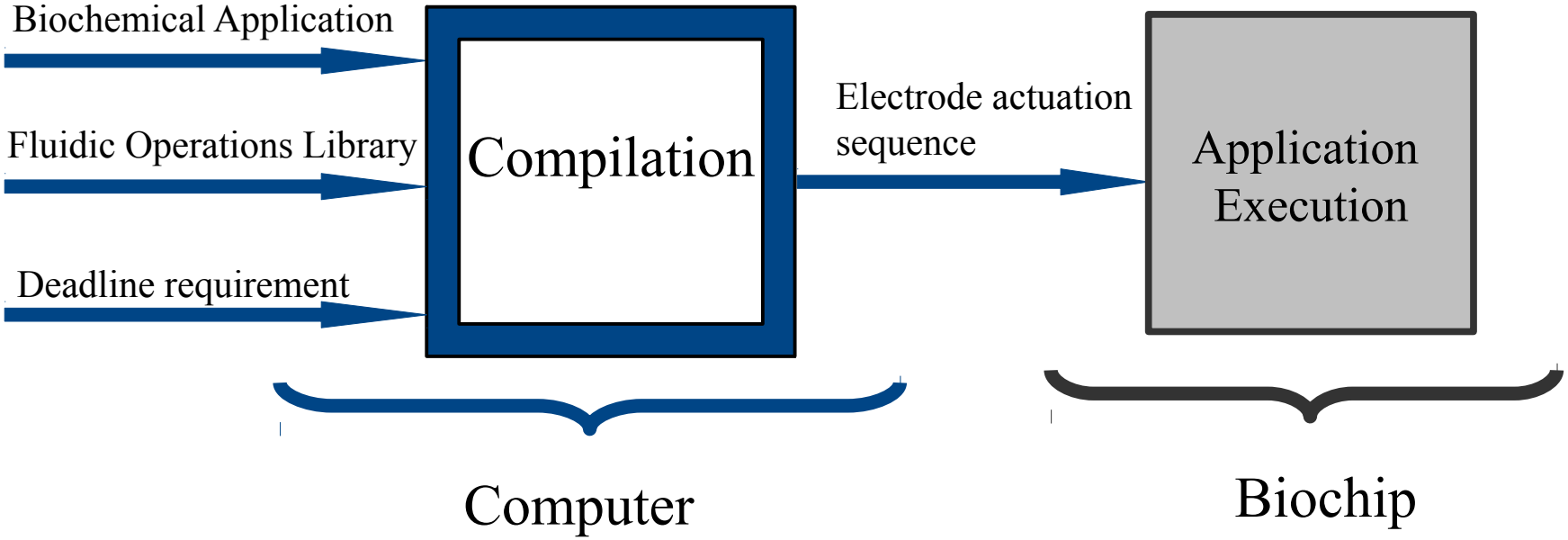
Operation	Area (cells)	Time (s)
Mix	2 x 2	10
Mix	2 x 3	6
Mix	1 x 4	5

$p^{90} = 0.1\%$

$p^{0} = 0.29\%$ $p^{00} = 0.58\%$

$p^{180} = -0.5\%$

Compilation Flow

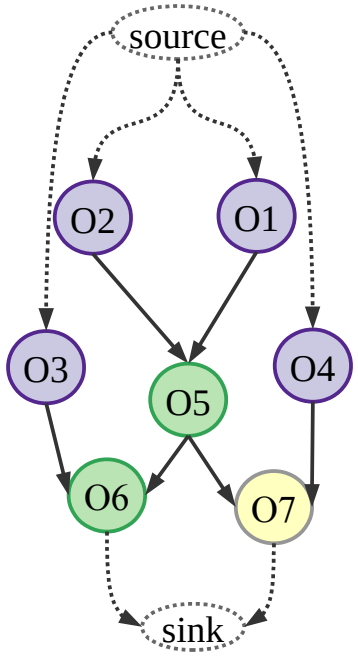


Compilation: Main steps

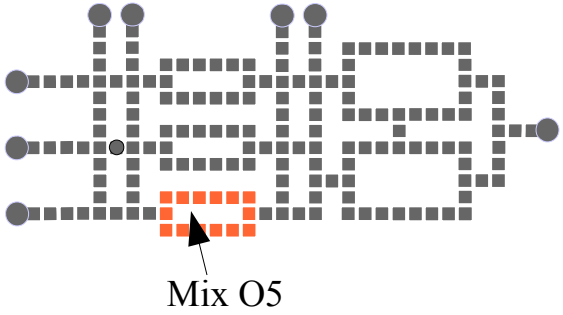
Allocation

Operation	Area	Time (s)
Mix	2x5	2
Mix	2x4	3
Mix	2x2	10
Detection	1x1	30

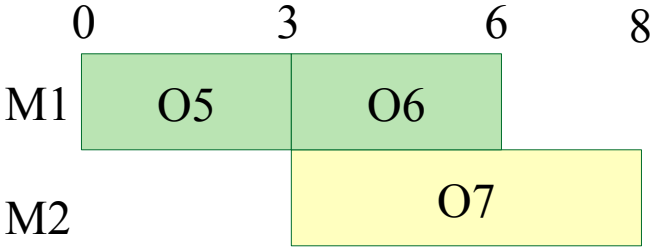
Binding



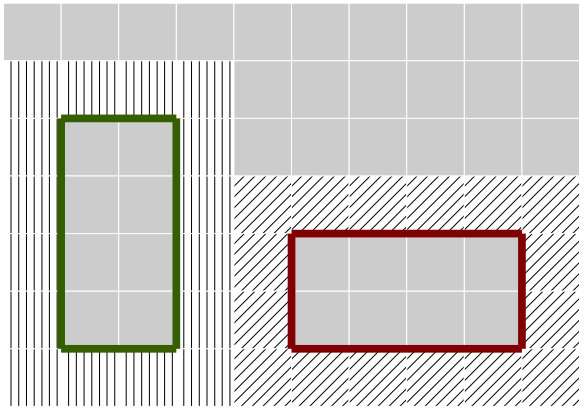
Placement

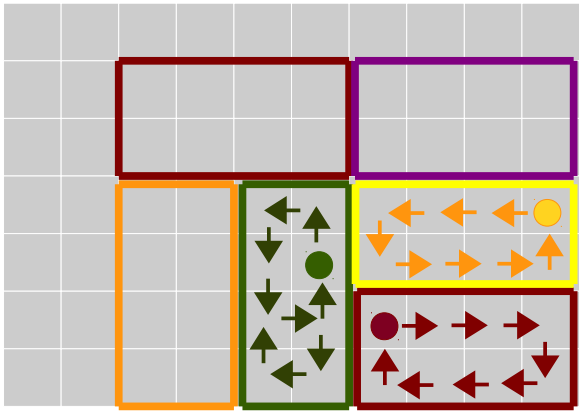


Scheduling



Droplet vs. Module Compilation





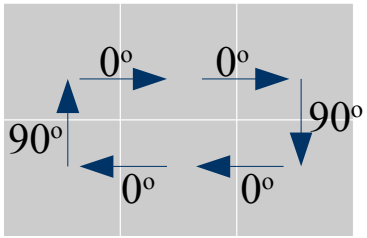
Module based

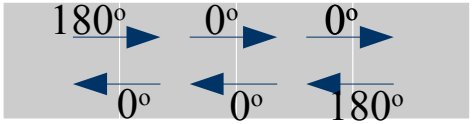
- module library
- black boxes
- protection borders

Droplet based

- routing base operation execution
- the position of the droplet is tracked
- better use of space

Operation	Area (cells)	Time (s)
Mix	2 x 4	3
Mix	2 x 2	4
Dilution	2 x 4	4





Libraries

Component Library

Name	Unit cost	Dimensions (mm)	Time (s)
Electrode	1	1.5×1.5	N/A
Input Reservoir	3	1.5×4.5	2
Waste Reservoir	3	1.5×4.5	N/A
Capacitive Sensor	1	1.5×4.5	0
Optical Detector	9	4.5×4.5	8

Virtual Devices Library

Op.	Shape	Time (s) no faults	Time (s) $k=1$	Time (s) $k=2$
Mix	3×6	2.52	2.71	3.77
Mix	5×8	2.05	2.09	2.3
Mix	4×7	2.14	2.39	2.51
Mix	5×5	2.19	2.28	2.71
Mix	$5\times5\times1$	2.19	2.73	3.92
Mix	$5\times5\times2$	3.98	5.82	7.56
Dilution	3×6	4.4	4.67	4.11
Dilution	5×8	3.75	4.76	6.3
Dilution	4×7	3.88	4.22	4.46
Dilution	5×5	3.98	4.12	4.67
Split	1×1	0	0	0
Storage	1×1	N/A	N/A	N/A

Test for k-vertex connectivity

Algorithm 1:

(1) For every i and j such that $1 \leq i < j \leq k$ check whether there are k vertex-disjoint paths between them. If for some i and j the test fails, then G's connectivity is less than k.

(2) For every j, $k+1 \leq j \leq n$, form $\tilde{G}$ and check whether there are k vertex-disjoint paths between a and j. If for some j the test fails, then G's connectivity is less than k.

(3) The connectivity of G is at least k.

Complexity $O(kn^3)$

Capacitive Sensor

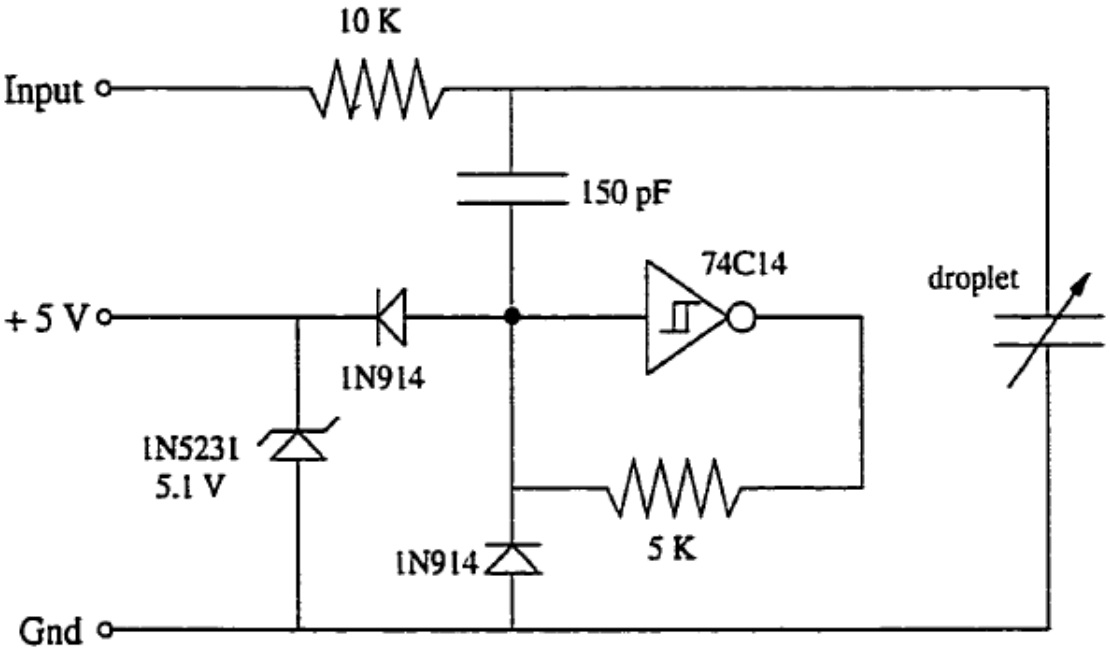


Figure 5.7: Ring-oscillator circuit for capacitance measurements of droplets [Pollack*]

Biochip Parameters

Parameter	Range or conditions
Gap height, h	0.19 – 1.04 mm, with $L = 1.5$ mm and $D/L \approx 1.3$
Droplet diameter, D	1.31 – 2.18 mm, with $L = 1.5$ mm and $L/h \approx 5$
Electrode pitch, L	0.15 – 1.50 mm, with $L/h \approx 5$ and $D/L \approx 1.3$
Electrode shape	interdigitated, square, circular, wedge
Medium type	air, silicone oil
Droplet viscosity	1.0 – 58.4 cP
Oil viscosity	0.8 – 9.4 cP
Droplet volume, V_D	3 nl – 3 μl
Velocity, v	0 – 10 cm/s

Table 4.1: Parameter space for droplet dynamic investigations [Pollack*]

Glucose detection

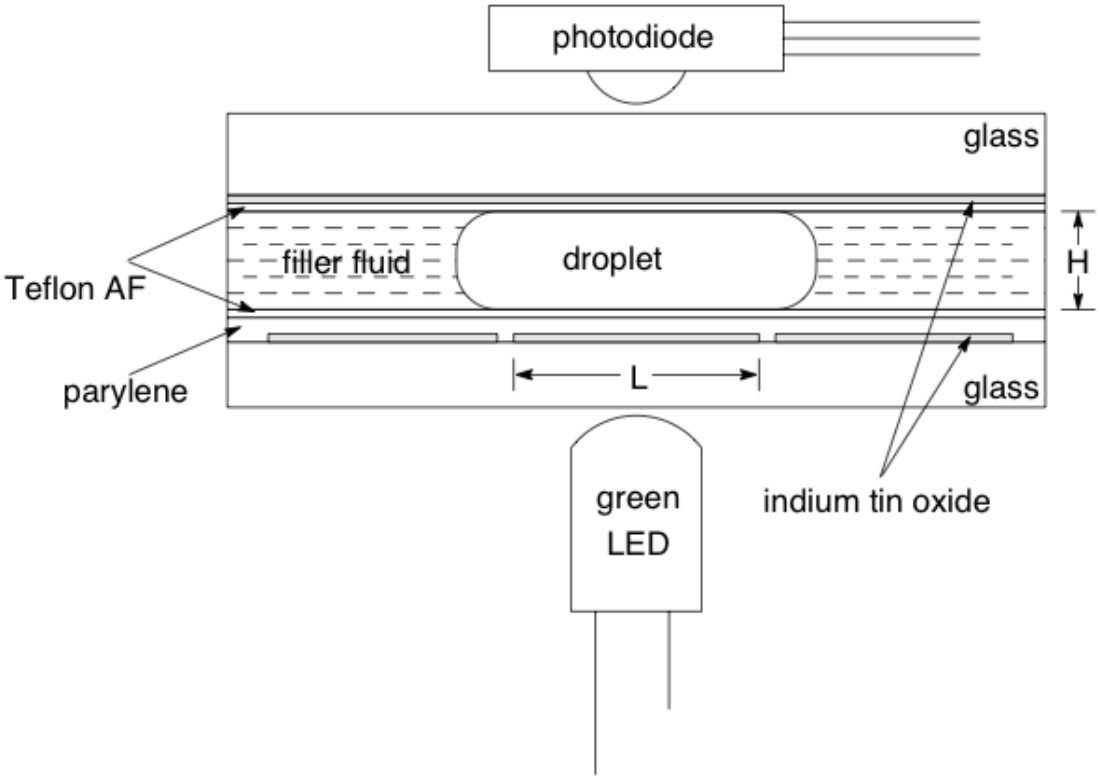
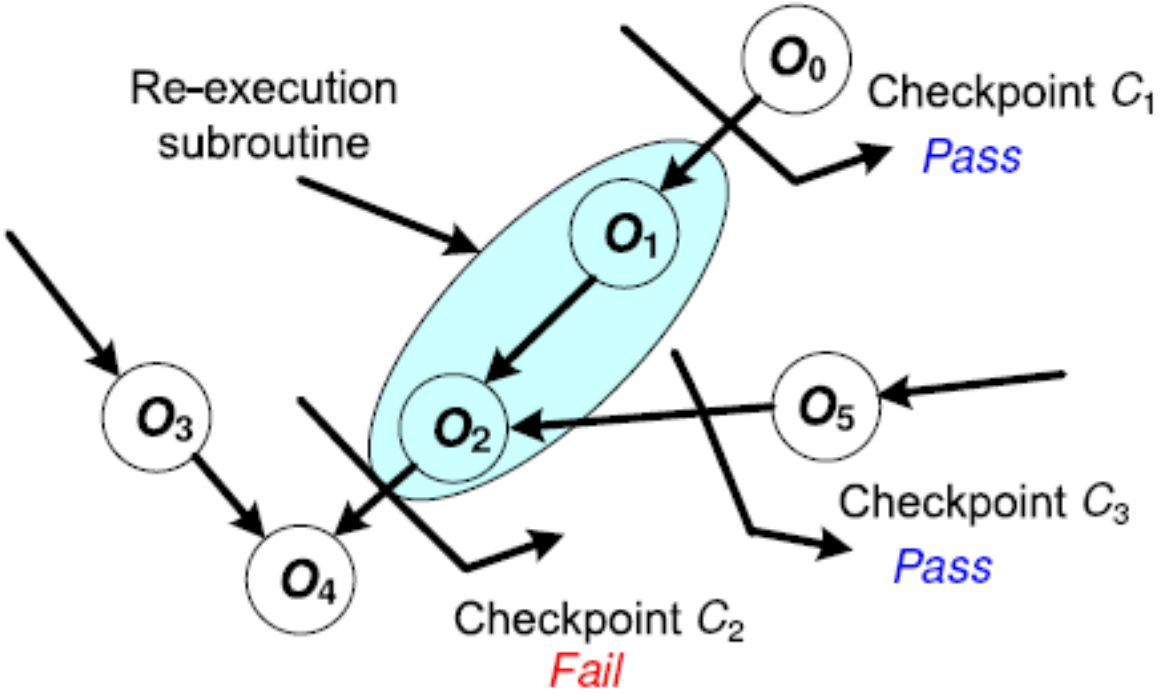


Fig. 2 Setup for glucose detection

Srinivasan 2003, Clinical diagnostics on human whole blood, plasma, serum, urine, saliva, sweat, and tears on a digital microfluidic platform

Control-Path Design

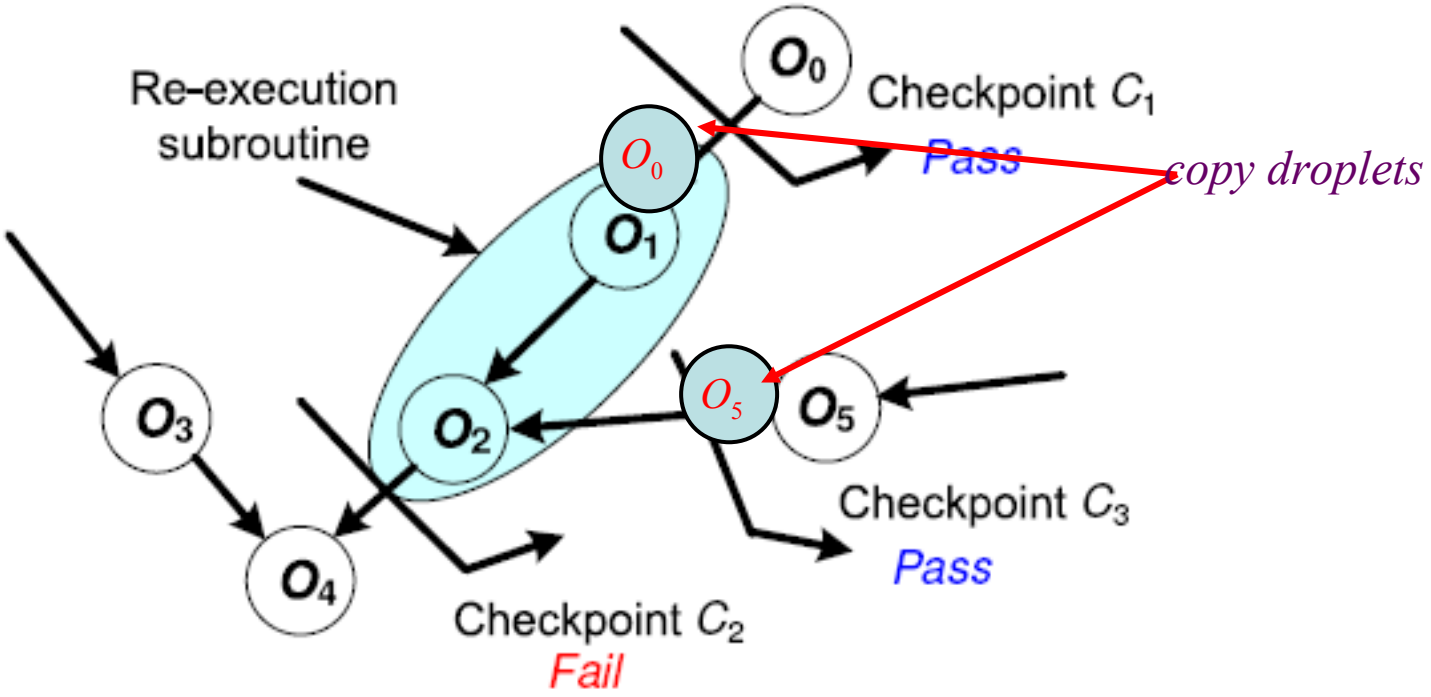
- Add *checkpoints* to monitor outcomes of fluidic operations
  - **Checkpoint**: storage of the intermediate product droplet
- Assign each checkpoint a *re-execution subroutine*
  - **Subroutine**: fluidic operations between checkpoints



Figures taken from Krish Chakrabarty, Duke University

Control-Path Design

- Extra **copy droplets** needed
- **Checkpoints**: where ?
- Costs:
  - Time
  - Area



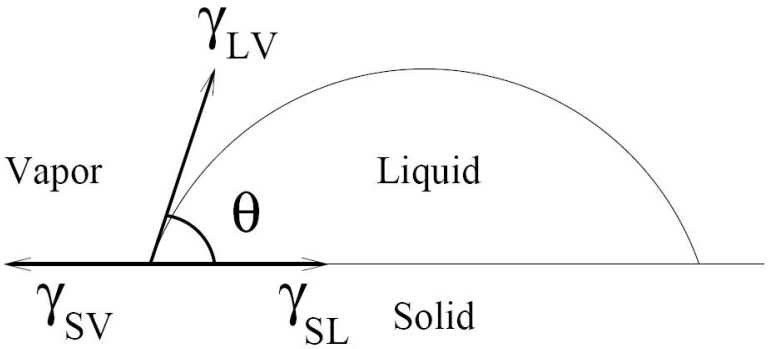
Figures taken from Krish Chakrabarty, Duke University

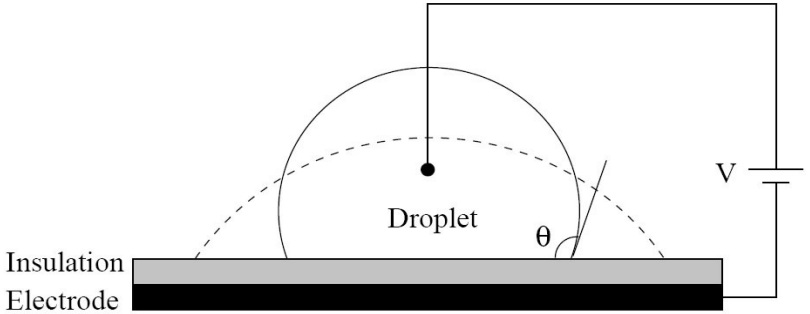
And the droplet moves!

- Electrowetting on dielectric principle (EWOD)
  - Electrical modulation of the solid-liquid interfacial tension

Young equation

$$\gamma_{LV}\cos(\theta) = \gamma_{SV} - \gamma_{SL}$$

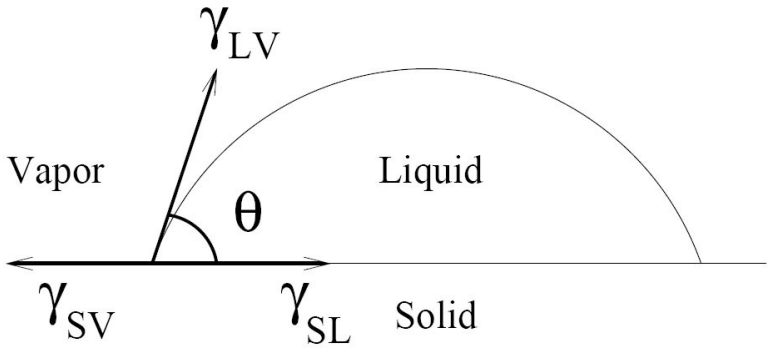




Figures taken from Krish Chakrabarty, Duke University

Electrowetting: Physical Principles (I)

- Motion of droplets is based on the differences between contact angles in the advancing and receding lines of a droplet.
- When a droplet rests on a non-wetting solid surface, the forces acting at the solid-liquid-vapor interface equilibrate and result in a contact angle θ between the droplet and solid, as described by Young's equation,



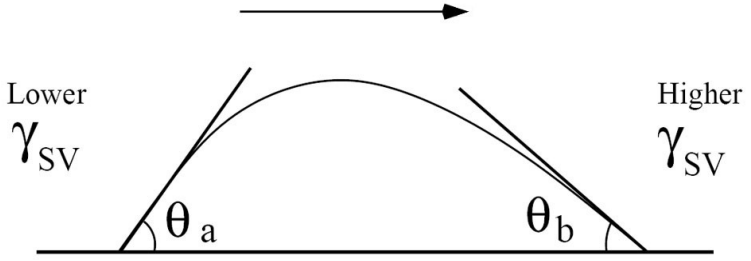
$$\gamma_{LV}\cos(\theta) = \gamma_{SV} - \gamma_{SL}$$

γ_{LV}, γ_{SV} and γ_{SL} are the liquid-vapor, solid-vapor and solid-liquid surface energies

Slide taken from Krish Chakrabarty, Duke University

Electrowetting: Physical Principles (II)

- When an imbalance in these surface energies occurs (as in the case of a droplet resting on a surface with a gradient surface energy), a net force is induced
  - Initiate droplet motion
- Imbalance can be induced by chemical, thermal, or electrostatic means
  - In the case of thermally-induced droplet motion, a surface tension gradient can be induced by differentially heating the ends of a droplet, since the surface tension of a liquid decreases with temperature.

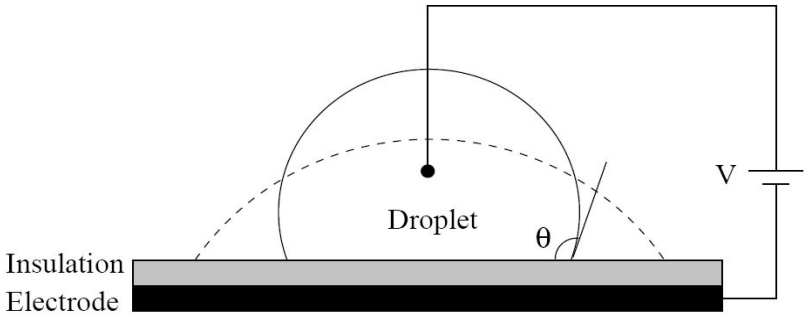


Electrowetting: Physical Principles (III)

- Electrowetting-based actuation of droplets: electrical fields used to induce surface tension gradients.
  - Electrowetting effect ⇒the surface energy can be directly modified by the application of an electric field
- Consider a droplet resting on a electrode separated by a hydrophobic insulator
  - A potential is applied between the droplet and the electrode, resulting in a capacitive energy *E* stored in the insulator. The resulting energy is:

$$E = \frac{\varepsilon_0 \varepsilon_r A}{2d} V^2$$

$$\Rightarrow \gamma_{SL}(V) = \gamma_{SL}(0) - \frac{\varepsilon_0 \varepsilon_r A}{2d} V^2$$



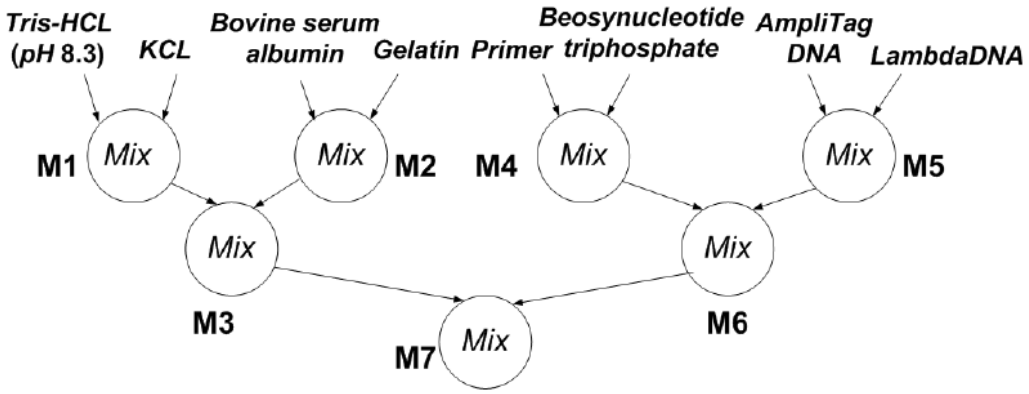
Contact angle change:

$$\cos\theta(V) = \cos\theta(0) - \frac{\varepsilon_0 \varepsilon_r A}{2d\gamma_{LV}} V^2$$

Reference: P. Y. Paik, V. K. Pamula and K. Chakrabarty,
"Adaptive Cooling of Integrated Circuits using Digital Microfluidics", Artech House, Norwood, MA, 2007.

Benchmarks: PCR

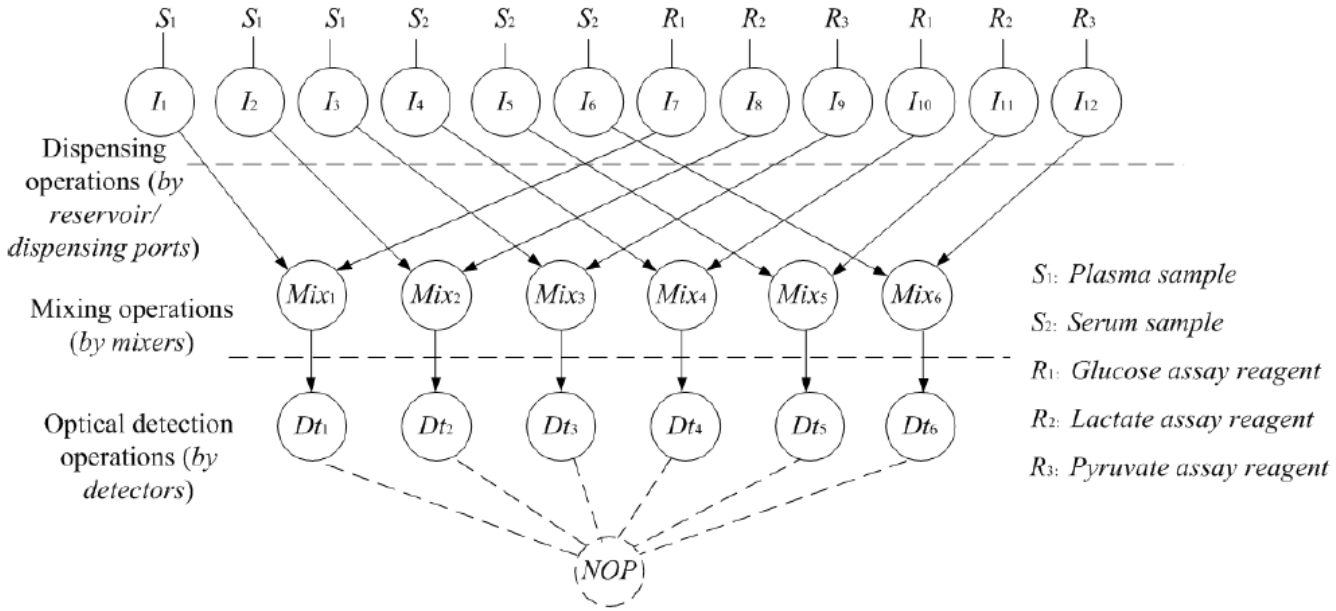
- Mixing stage for Polymerase Chain Reaction
- Electrode pitch: 1.5 mm, Gap height: 600 μm



Operation	Hardware*	Module	Mixing time
mixing	2x2 electrode array	4x4 cells	10s
	4-electrode linear array	3x6 cells	5s
	2x3 electrode array	4x5 cells	6s
	2x4 electrode array	4x6 cells	3s

Benchmarks: IVD

- Multiplexed in-vitro diagnosis
- Electrode pitch: 1.5 mm, Gap height: 600 μm



Benchmarks: CPA

- Colorimetric Protein Assay
- Electrode pitch: 1.5 mm, Gap height: 600 µm

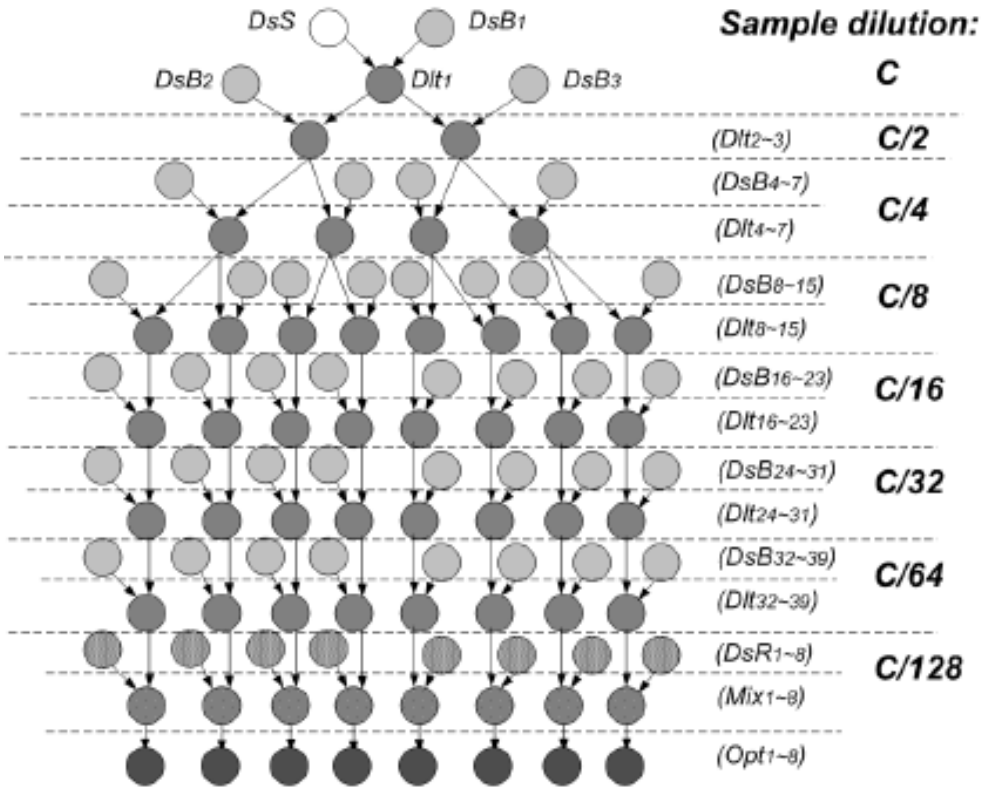


Figure 3-2. Sequencing graph for a protein assay.