Programmable Microfluidics

William Thies*, J.P. Urbanski†,
Mats Cooper†, David Wentzlaff*,
Todd Thorsen†, and Saman Amarasinghe*

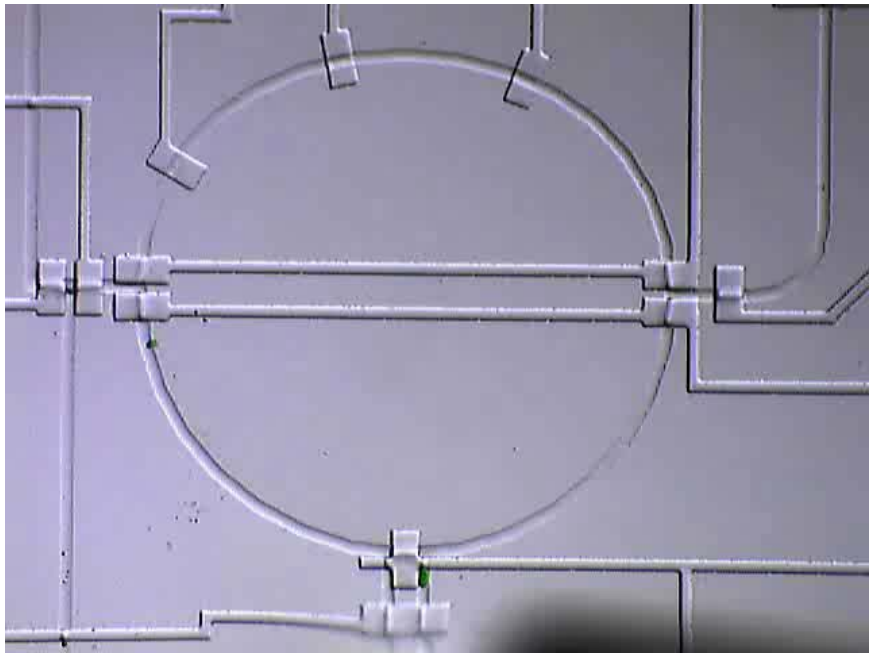
* Computer Science and Artificial Intelligence Laboratory
† Hatsopoulos Microfluids Laboratory

Massachusetts Institute of Technology

October 12, 2004

Microfluidic Chips

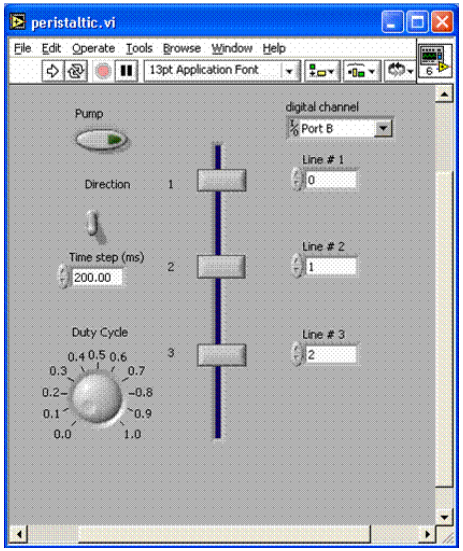
- Idea: a whole biological lab on a single chip
  - Input/output
  - Actuators: temperature, light/dark, cell lysis, etc.
  - Sensors: luminescence, pH, glucose, etc.

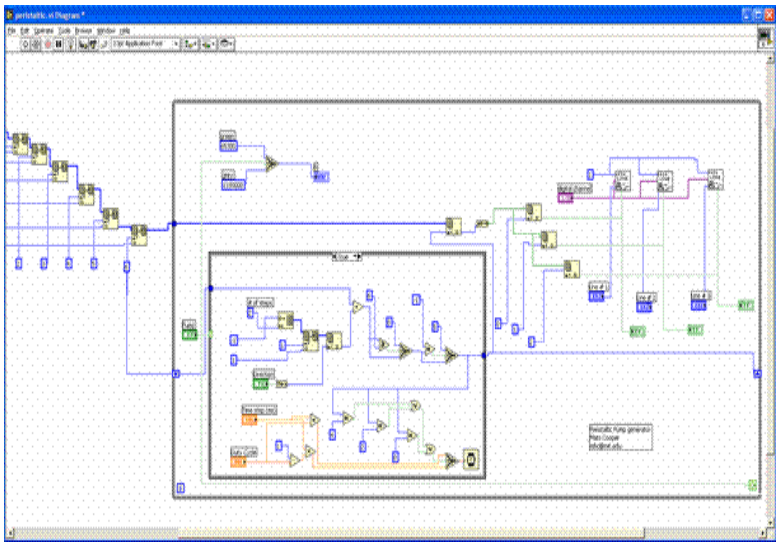


- Benefits:
  - Small sample volumes
  - High throughput
  - Geometrical manipulation

Our Goal:
Provide Abstraction Layers for this Domain

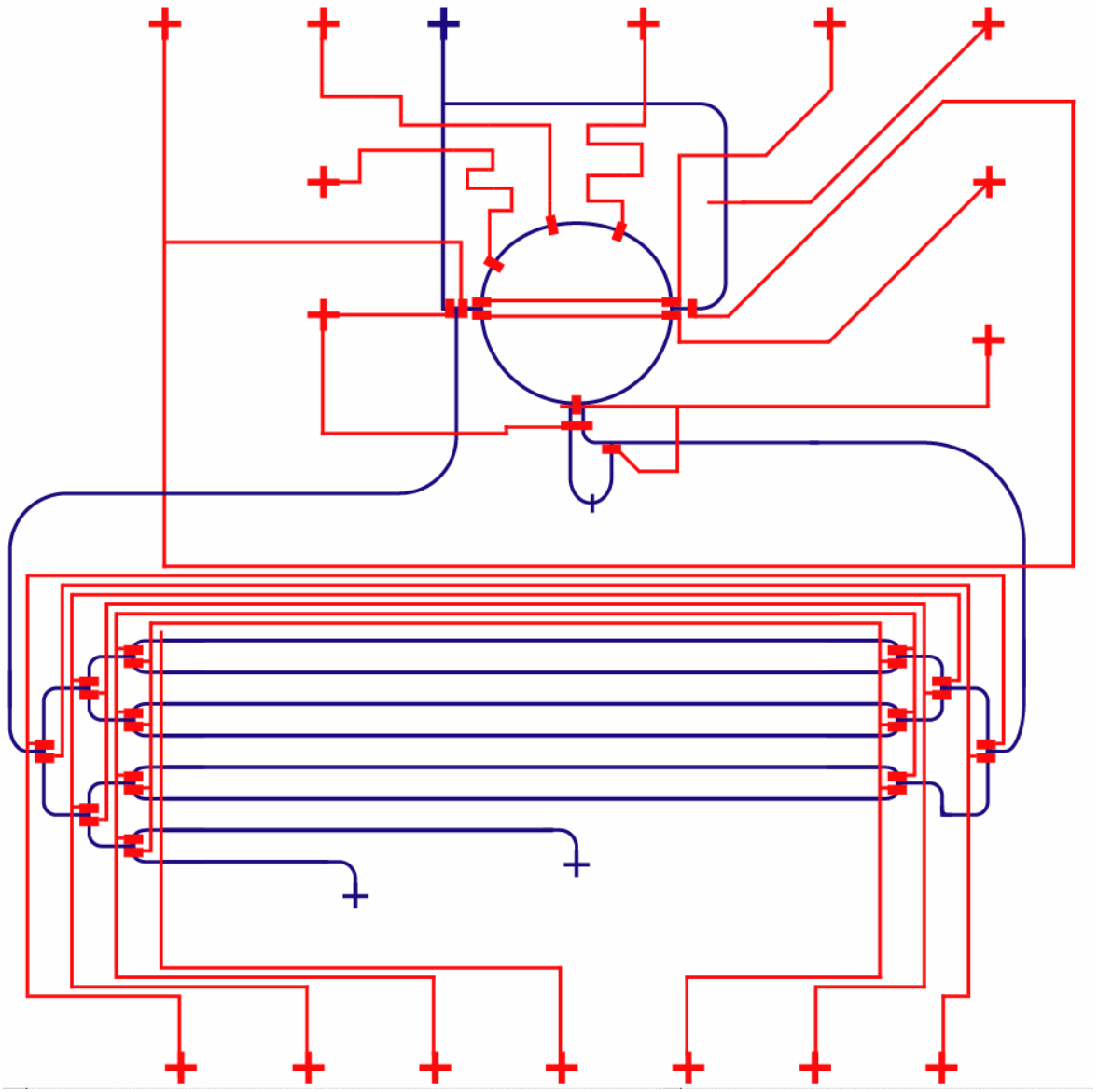
- Current interface: gate-level control (Labview)





- New abstraction layers will enable:
  - Scalability
  - Adaptivity
  - Portability
  - Optimization
- NOT our goal: replace silicon computation

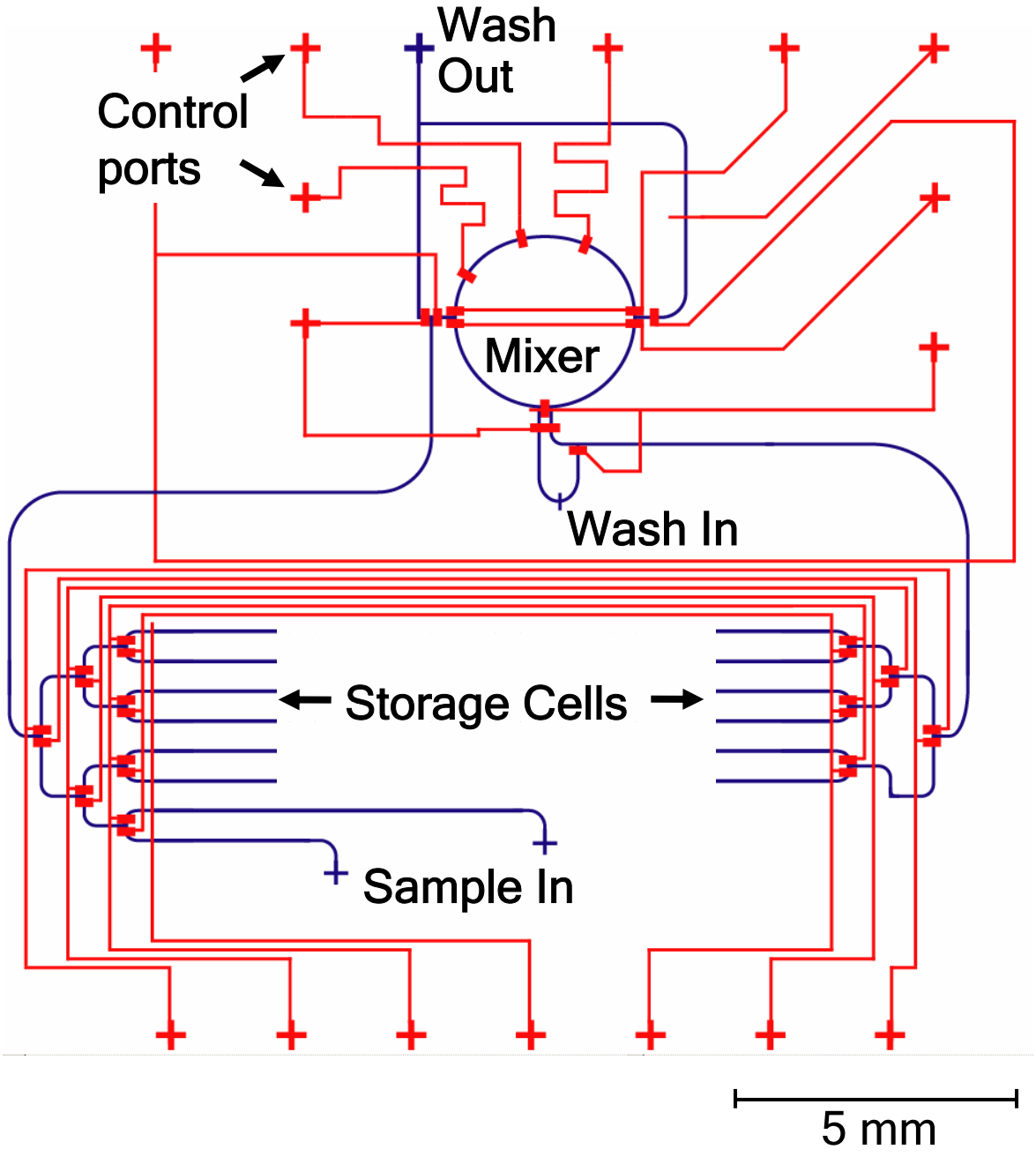
A General-Purpose Microfluidic Chip



Control layer
Flow layer

5 mm

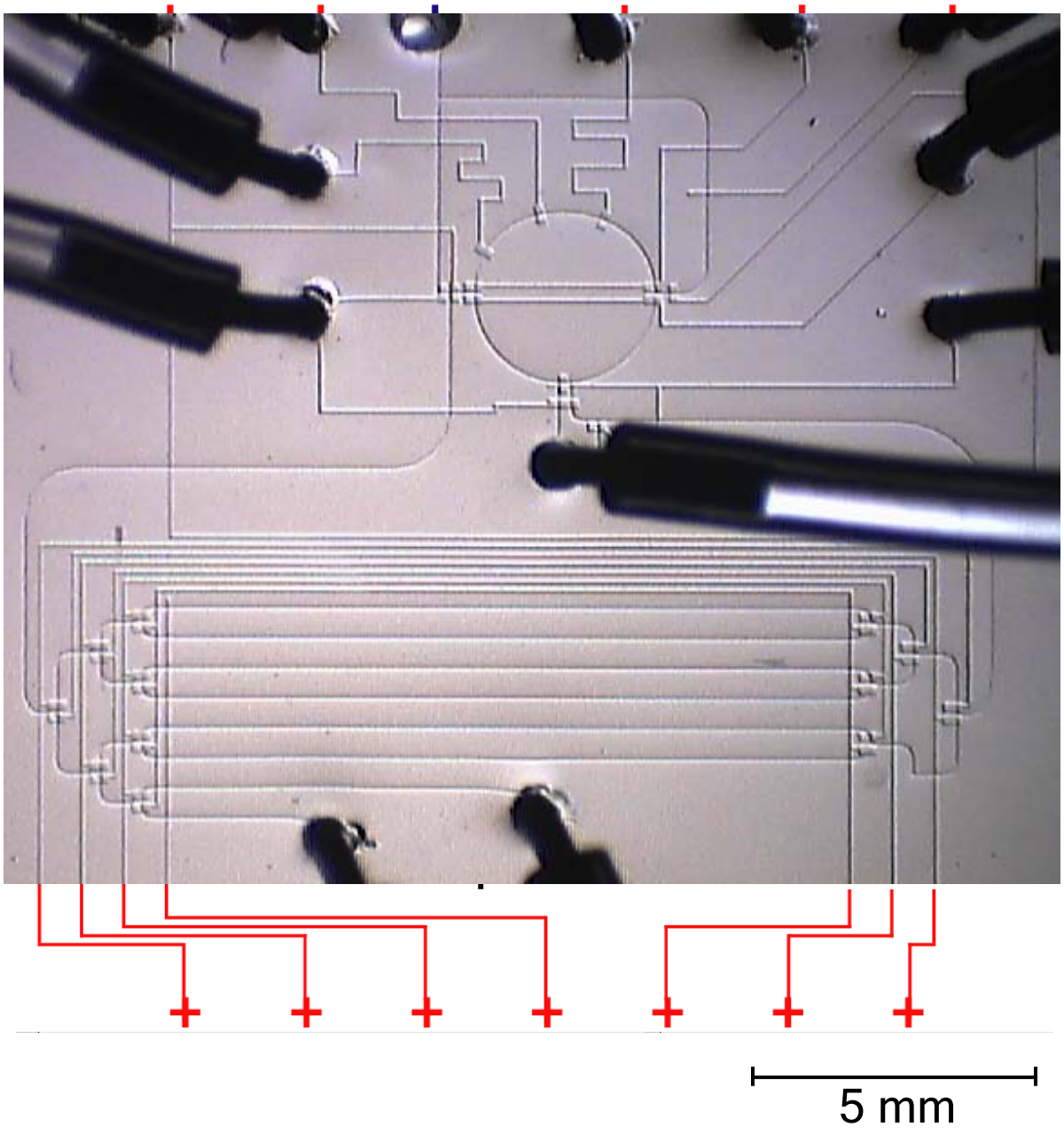
A General-Purpose Microfluidic Chip



Control layer

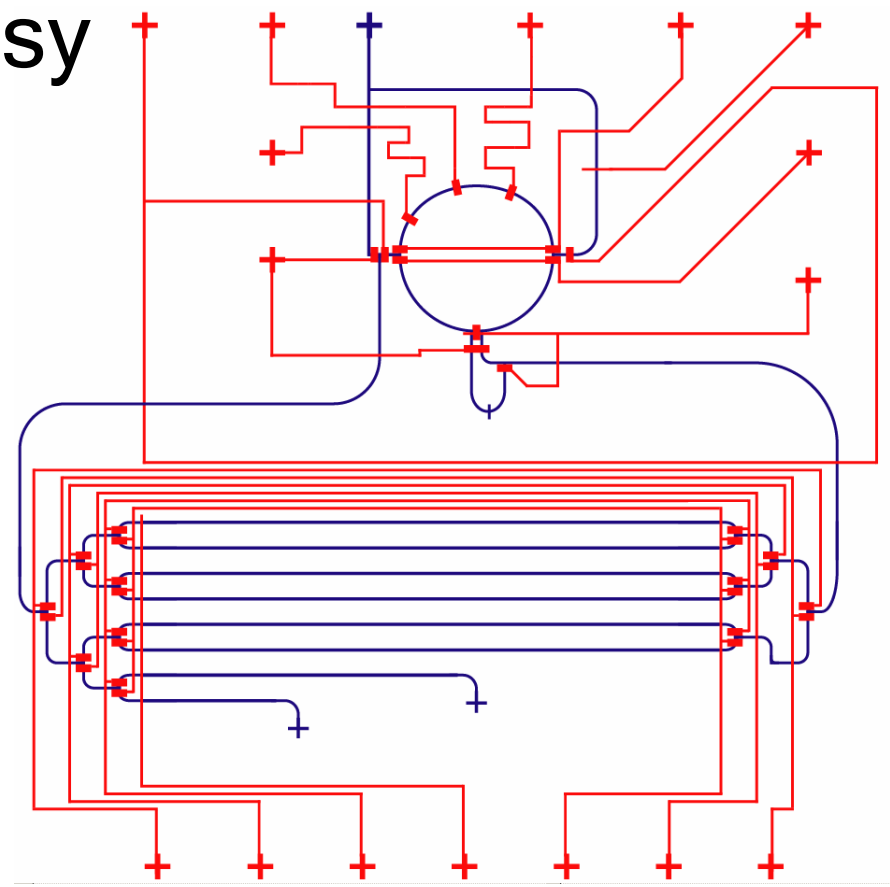
Flow layer

A General-Purpose Microfluidic Chip



Providing a Digital Abstraction

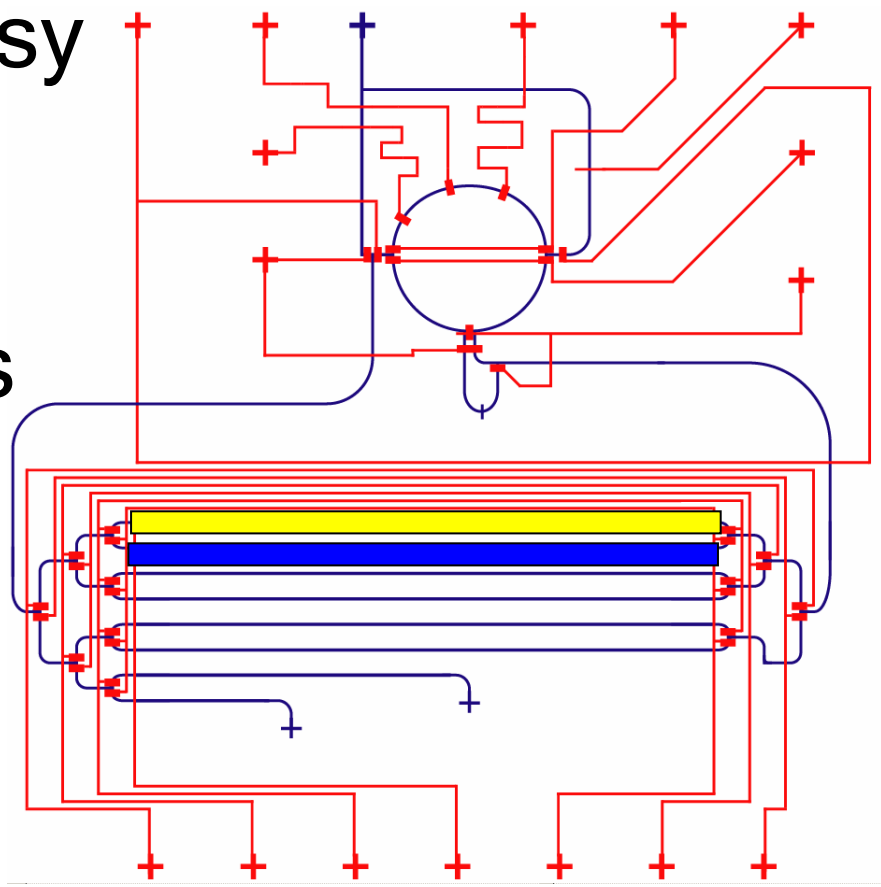
- All fluid operations are lossy
- How to control the error?



Providing a Digital Abstraction

- All fluid operations are lossy
- How to control the error?

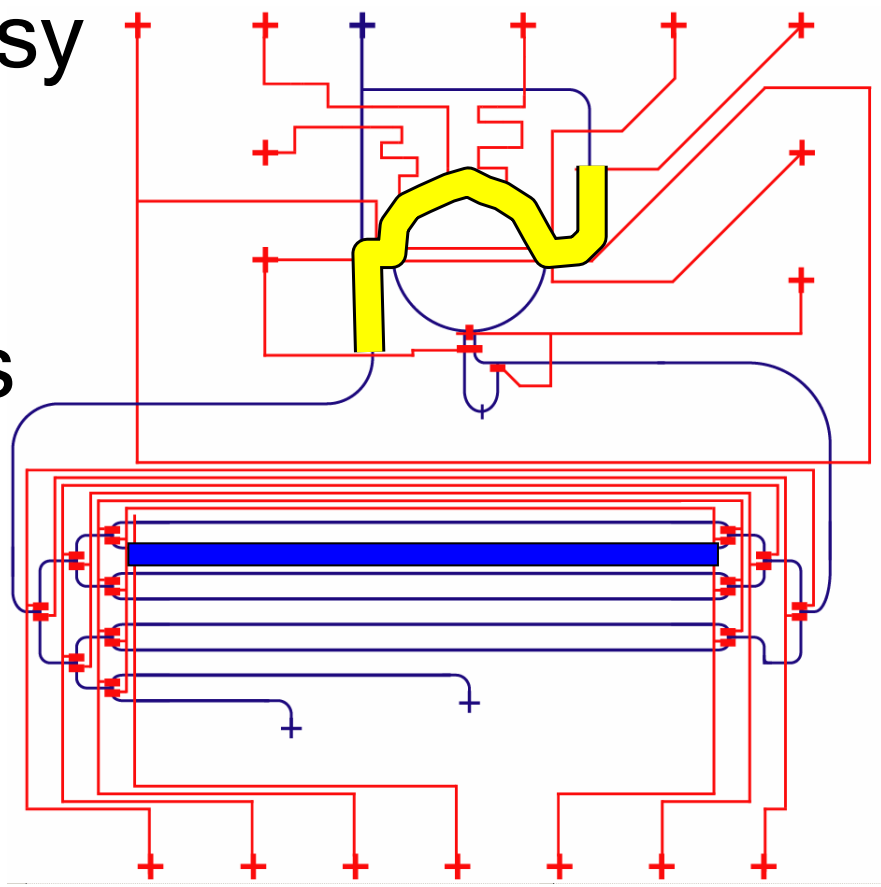
- Solution: discrete samples
  - throw out half of sample on each mix



Providing a Digital Abstraction

- All fluid operations are lossy
- How to control the error?

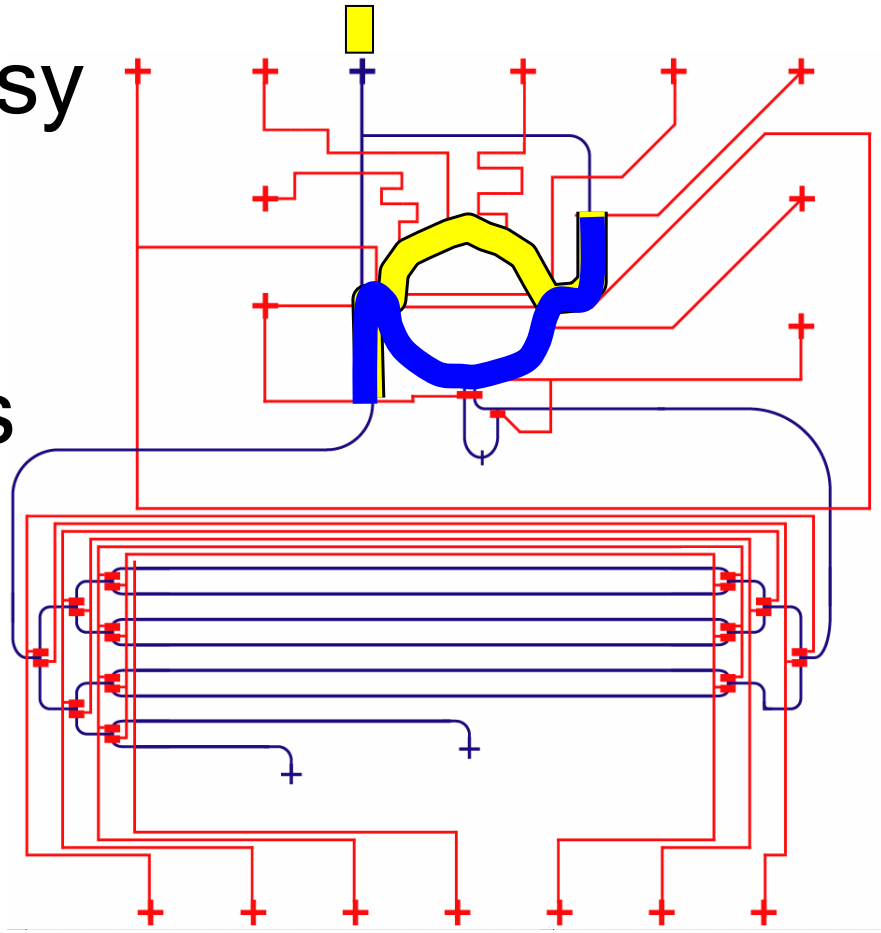
- Solution: discrete samples
  - throw out half of sample on each mix



Providing a Digital Abstraction

- All fluid operations are lossy
- How to control the error?

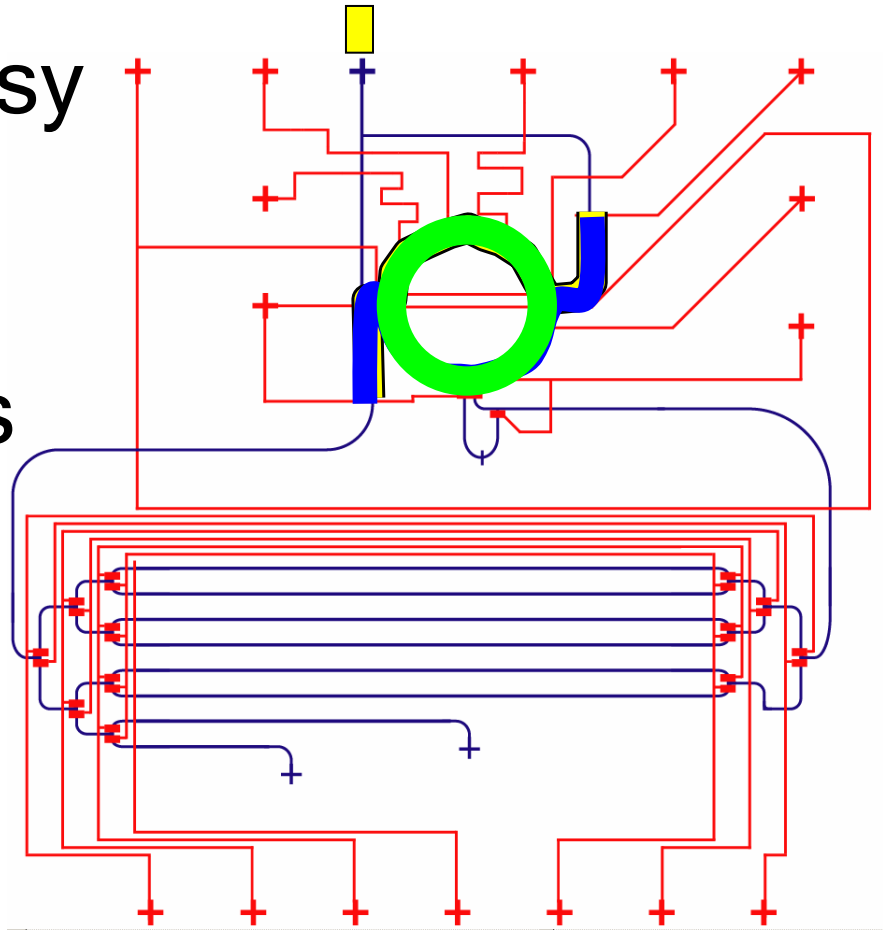
- Solution: discrete samples
  - throw out half of sample on each mix



Providing a Digital Abstraction

- All fluid operations are lossy
- How to control the error?

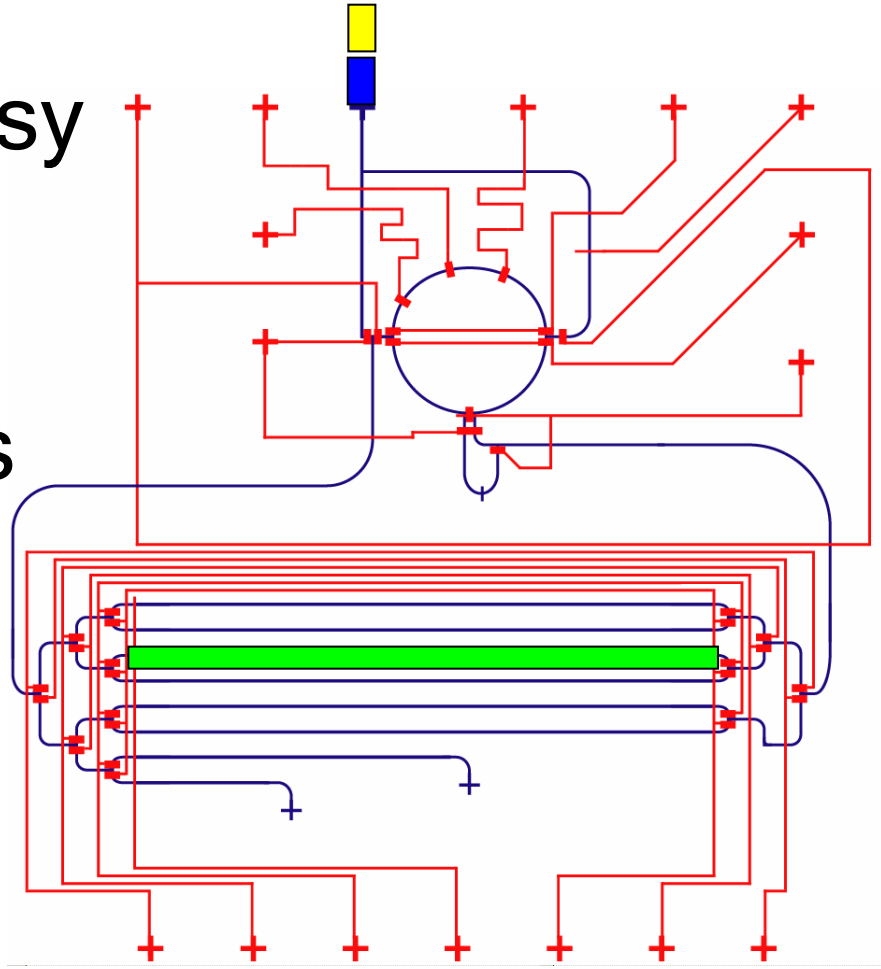
- Solution: discrete samples
  - throw out half of sample on each mix



Providing a Digital Abstraction

- All fluid operations are lossy
- How to control the error?

- Solution: discrete samples
  - throw out half of sample on each mix



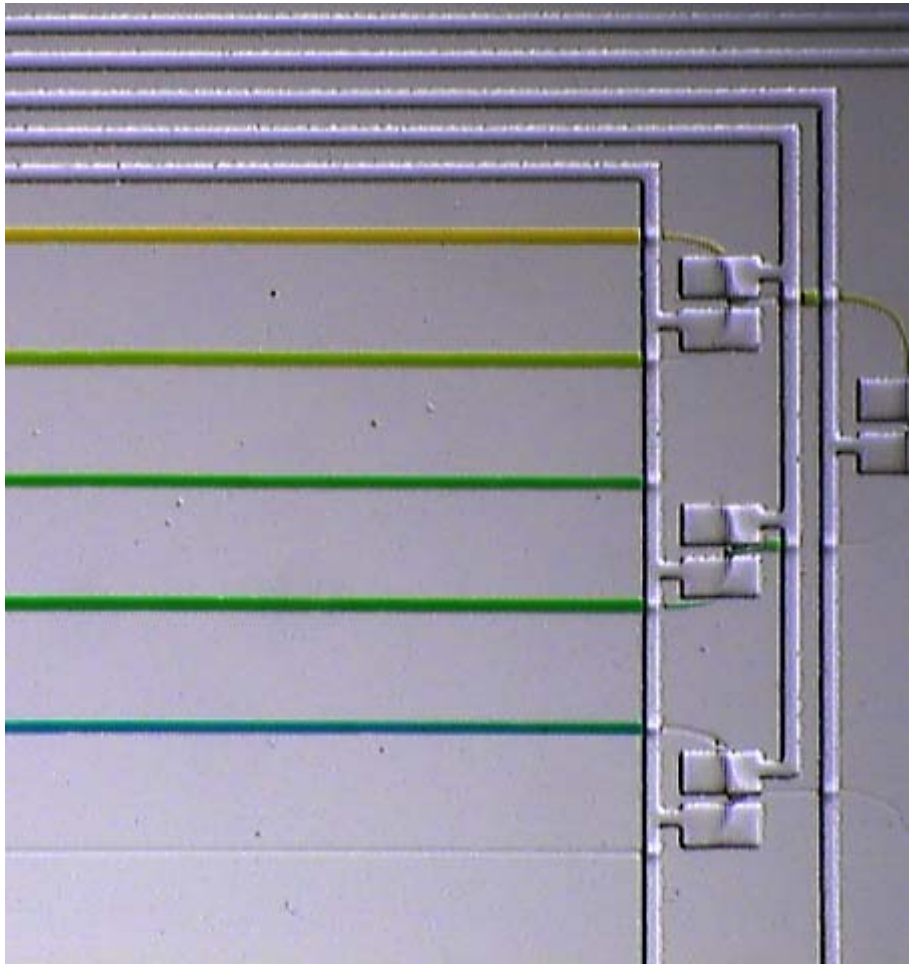
Programming Model

```
Fluid blue = input (0);
Fluid yellow = input(1);
for (int i=0; i<=4; i++) {
    mix(blue, i/4, yellow, 1-i/4);
}
```

New abstractions:

- Regenerating fluids

- Efficient mixing algorithms



450 Valve Operations

Example: Fixed pH Reaction

```
Fluid sample = input (0);
Fluid acid = input(1);
Fluid base = input(2);
do {
    // test pH of sample
    Fluid pH_test = mix(sample, 0.9, indicator, 0.1);
    double pH = test_luminescence(pH_test);
    // if pH is out of range, adjust sample
    if (pH > 7.5) {
        sample = mix (sample, 0.9, acid, 0.1);
    } else if (pH < 6.5) {
        sample = mix (sample, 0.9, base, 0.1);
    }
    wait(5);
} while (detect_activity(sample));
```

Example: Fixed pH Reaction

```
Fluid sample = input (0);
Fluid acid = input(1);
Fluid base = input(2);
do {
    // test pH of sample
    Fluid pH_test = mix(sample, 0.9, indicator, 0.1);
    double pH = test_luminescence(pH_test);
    // if pH is out of range, adjust sample
    if (pH > 7.5) {
        sample = mix (sample, 0.9, acid, 0.1);
    } else if (pH < 6.5) {
        sample = mix (sample, 0.9, base, 0.1);
    }
    wait(5);
} while (detect_activity(sample));
```

Feedback-Intensive Applications:

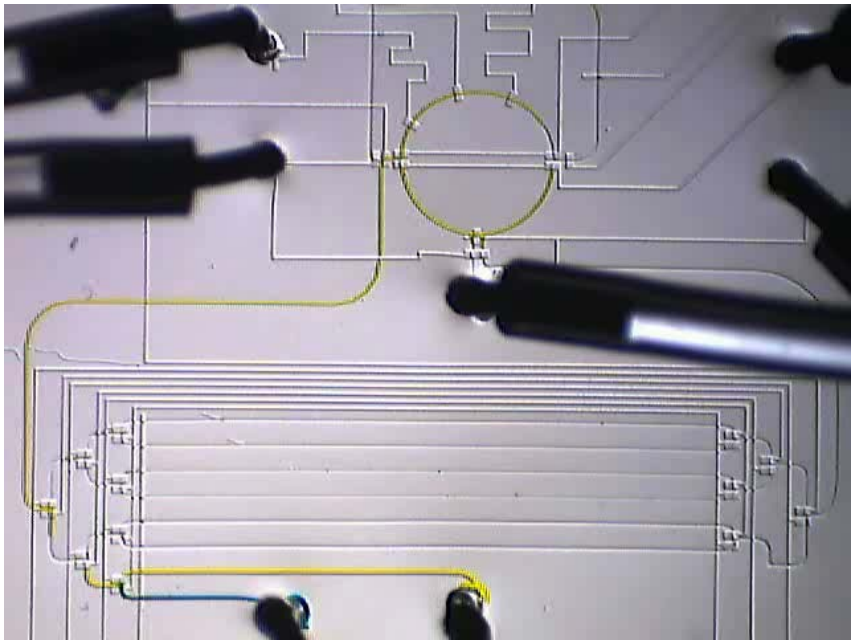
- Cell isolation and manipulation
- Dose-response curves
- High-throughput screening
- Long, complex protocols

Opportunities for Computer Scientists

- Experimental biology is becoming a digital science
  - What are the right abstraction layers?
  - We can have a large impact

Software:
- scheduling
- programming abstractions
- verifying safety properties
- optimizing throughput, cost

Hardware:
- parallelism
- error tolerance
- reducing design complexity
- minimizing control overhead



- Vision: A defacto language for experimental science