A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

B.Sc. (Honours with Research)

in

Computer Science

HyDRA: A Slug-Flow Microfluidic Architecture and Instruction Set

by

Aarush Kumbhakern
1020221652

under the supervision of

Prof. Bhargab B. Bhattacharya

Department of Computer Science
Ashoka University

April 2026

Declaration

I hereby declare that the work presented in this thesis has been carried out by me under the supervision of Prof. Bhargab B. Bhattacharya at Ashoka University. The material presented is the outcome of my own original research, except where explicitly acknowledged through citation to the published literature. No portion of this work has been submitted for any other degree or diploma at this or any other institution.

Aarush Kumbhakern	Date
1020221652	

Certificate

This is to certify that the thesis titled *HyDRA: A Slug-Flow Microfluidic Architecture and Instruction Set*, submitted by Aarush Kumbhakern (1020221652) in partial fulfillment of the requirements for the degree of B.Sc. (Honours with Research) in Computer Science at Ashoka University, is a bona fide record of the research work carried out by him under my supervision. The contents of this thesis, in full or in part, have not been submitted to any other institute or university for the award of any degree or diploma.

Prof. Bhargab B. Bhattacharya
Department of Computer Science
Ashoka University

Acknowledgments

I owe my deepest gratitude to my supervisor, Prof. Bhargab B. Bhattacharya, whose guidance, insights, and patience shaped both this work and my understanding of what it means to do research. The architectural discipline of separating abstraction layers, the insistence on quantitative physical grounding, and the conviction that formal rigor and physical realizability are complementary rather than competing are lessons that run throughout the thesis.

I thank the faculty and staff of the Department of Computer Science at Ashoka University for providing the academic environment in which this work was conceived. Conversations with peers working on adjacent problems in design automation and microfluidics helped sharpen many of the arguments that appear in the later chapters.

Finally, I thank my family for their unwavering support throughout my undergraduate years.

Abstract

Programmable microfluidic biochips have matured as a fabrication platform but lag as a computing platform: continuous-flow chips expose dozens of individually-addressable valves driven by layout-specific, hand-written sequences, and higher-level abstractions either sacrifice substrate generality or cannot offer strong static guarantees because the underlying fluid state is continuous and dispersion-coupled.

This thesis presents **HyDRA**, a programmable microfluidic architecture that exposes aqueous-slug handling as a discrete, integer-valued instruction set. The substrate is a closed loop of two-phase slug flow served by a single peristaltic pump and ten register tiles, realizable in standard multilayer PDMS with Quake-style pneumatic valves. Ten slug operations and two control-flow constructs, managed under a single-static-assignment discipline, form the user-visible ISA; the underlying valves, registers, and pump strokes are managed by a compiler.

The thesis develops HyDRA along four technical contributions. First, a set of architectural principles:

- volume quantization by the pump quantum,
- an always-full sealed-loop hydraulic transmission,
- and a single closed active flow path

reduces a continuous physical system to a discrete integer state space. Second, a physical feasibility argument shows that the standard HFE-7500 / sub-CMC PFPE-PEG / Aquapel-treated PDMS fluid system admits an operating window in which both wall-film stability and on-demand coalescence hold simultaneously, with HyDRA's parameters sitting inside this window with margin. Third, a slug-functional ISA is specified, with native asynchronous module contracts that recover pipeline parallelism on an otherwise serial-bus architecture. Fourth, a finite-state abstraction of the chip supports eight invariants, each a Boolean combination of integer linear inequalities, making every safety-relevant property decidable ahead of time at cost polynomial in program size and live-slug count for bounded branch depth.

Every architectural quantity — pump quantum, slug regime, coalescence timescale, valve count — is anchored in the published microfluidic literature. Fabrication, measured evaluation, and a reference compiler are identified as follow-up work.

A condensed version of this thesis has been submitted to the 2026 IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2026).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Microfluidic biochips, molded into PDMS at micrometer resolution, have matured over the past two decades as a fabrication platform. Multilayer soft-lithographic PDMS with Quake-style pneumatic valves [42] routinely yields chips with hundreds to thousands of valves through standard photolithography.

What has *not* matured is programmability. A typical continuous-flow chip exposes dozens of individually-addressable valves driven by hand-written pneumatic sequences tied to a specific layout; reprogramming for a different assay frequently requires redesigning both the sequence and the layout. Higher-level abstractions exist but fall into one of two camps: substrate-agnostic protocol languages, which carry no substrate model to reason about; or substrate-specific architectures, whose static guarantees are limited by the substrate's physics — the dispersion-coupled concentration field of continuous-flow chips, or the stochastic actuation of EWOD electrodes.

The central claim of this thesis is that an instruction-set-style abstraction *is* achievable for microfluidic biochips, *provided the fluid substrate is chosen with that abstraction in mind.* Two-phase slug flow makes every volume an integer multiple of a pump quantum, every position an integer cell index up to bounded drift that periodic re-registration zeroes, and every operation a finite composition of pump strokes and valve switches — reducing chip behavior to a finite, integer-valued state space that supports compile-time verification and a cleanly layered software stack.

1.2 HyDRA: An Overview

HyDRA (High-level Discrete-slug bus-Register Architecture) is developed in four architectural layers:

1. **Two-phase regime.** Aqueous slugs transported through fluorocarbon carrier oil admit simultaneous wall-film stability during transport and on-demand coalescence at parked valves.

2. **Physical architecture.** A closed loop channel with ten register tiles, a mixing contour, an I/O zone, and a peristaltic pump, addressed through a shared pneumatic demultiplexer.

3. **Microarchitecture.** Five primitives for circulation, tile-mode set, mixer set, I/O set, and wait, into which every ISA operation lowers.

4. **Instruction set.** Ten slug operations plus two control-flow operations (`branch` and `repeat`), under a single-static-assignment (SSA) discipline.

Every HyDRA program is a finite, conditionally-branching sequence of these twelve operations over

named slugs. A compiler lowers each operation into microarchitectural primitives while tracking a small finite-state abstraction of the chip. Because the abstraction is finite and integer-valued, every safety-relevant property reduces to a Boolean combination of integer linear inequalities, decidable ahead of time at cost polynomial in program size and live-slug count for bounded branch depth.

1.3 Scope and Contributions

HyDRA is an architecture-specification contribution. The thesis *does not* include fabricated hardware, measured evaluation, a reference compiler, or a machine-checked proof — each is identified as a concrete continuation in Chapter 14. The physical-feasibility argument is conditional on a single per-chip calibration of the surfactant adsorption isotherm at the chosen geometry and temperature, performed by standard pendant-drop tensiometry. The specific contributions are summarized in Table 1.1.

Table 1.1: Contributions of the thesis.

	Contribution	Short description	Developed in
C1	Closed-loop slug-flow architecture	Realizable in standard multilayer PDMS, specified to the level of channel geometry, valve layout, and fabrication envelope	Chs. 5–7
C2	Physical-feasibility argument	The HFE-7500 / PFPE-PEG / Aquapel fluid system admits an operating window for wall-film stability and on-demand coalescence; HyDRA sits inside it	Ch. 6
C3	Five-primitive microarchitecture	Explicit mode-transition semantics and a re-registration discipline bridging the integer abstraction to physical reality	Ch. 9
C4	Slug-functional ISA	SSA discipline; native asynchronous module contracts that recover pipeline parallelism on a serial-bus architecture	Ch. 10
C5	Finite-state formal abstraction	Eight invariants, each a Boolean combination of integer linear inequalities; ahead-of-time verification, polynomial in program size and live-slug count for bounded branch depth	Ch. 8
C6	Coverage argument	Target assay class — PCR, immunoassay, serial dilution, protein crystallization, and glucose assay — expressible within HyDRA's resources	Ch. 11

1.4 Organization

Chapters 2–4 give background, literature survey, and problem statement. Chapters 5–11 develop the architecture in dependency order: principles, operating regime, physical architecture, formal model, microarchitecture, ISA, and coverage. Chapters 12–14 collect results, state conclusions, and identify follow-up work.

Chapter 2

Background

This chapter sets out the physical and fabrication vocabulary that the rest of the thesis draws on. A reader familiar with Quake-valve PDMS and slug-flow hydrodynamics may skim it.

2.1 Microfluidic Biochips

A *microfluidic biochip* is a planar device in which reagents are transported through micrometer-scale channels by pressure, electrowetting, electrophoretic, or capillary forces. Channel dimensions are small enough that $Re \ll 1$ throughout (laminar flow) but large enough for bulk-fluid continuum mechanics to apply. Three substrate families dominate.

Continuous-flow (CF). Reagent transport proceeds through pressure-driven flow in a network of channels gated by valves at junctions [41, 29, 42]. The fluid state at any instant is a continuous concentration field, coupled between adjacent segments by diffusion and Taylor–Aris dispersion. The substrate is mature: fabrication processes are well documented, valve fatigue is characterized, and pumping performance is reproducible to within a few percent [15]. Its limitation, from a programmability standpoint, is the continuous nature of the abstract state.

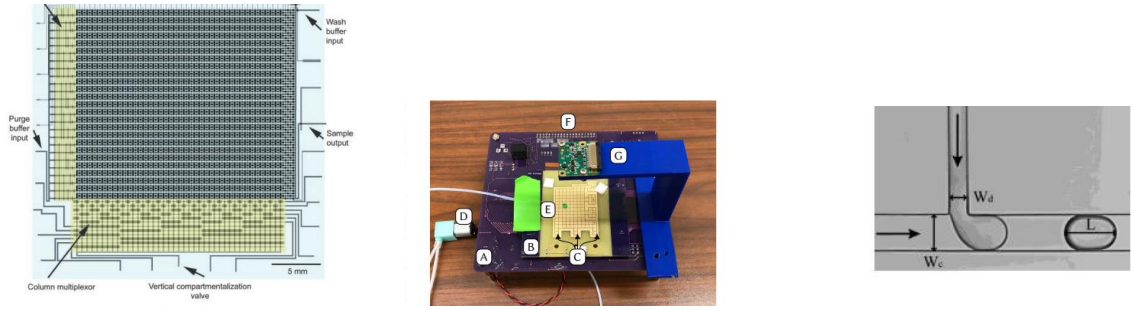
Electrowetting-on-dielectric (EWOD). Discrete aqueous droplets move across a 2D electrode grid by electrostatically modulating contact-angle asymmetry. Droplet identity is explicit and discrete, but per-electrode addressing limits channel density and forces substrate-specific control electronics. Integration with bulk microfluidic components (pumps, mixers, detectors) is non-trivial and tends to be substrate-coupled.

Two-phase slug flow (SF). An aqueous *slug* — a discrete plug of aqueous phase — travels along a one-dimensional channel encapsulated by a fluorocarbon carrier oil [6]. The channel remains continuous, the aqueous phase is discrete and identity-preserving, and addressing cost concentrates in the valves gating entry to storage and mixing regions. The substrate has been extensively characterized [14, 28, 10, 34].

HyDRA is built on the slug-flow family. Slug flow is the only substrate of the three to simultaneously offer (a) discrete aqueous identity, (b) a one-dimensional topology compatible with a closed-loop bus, and (c) a cross-section uniform enough to make pump displacement translate directly into a known column shift. These properties are what make an integer-valued instruction set physically meaningful, and they are what the rest of this chapter grounds in specific fabrication and hydrodynamic detail.

2.2 Soft-Lithographic PDMS and Quake Valves

The dominant fabrication platform is multilayer soft-lithographic PDMS with pneumatic valves, introduced by Unger *et al.* [42]. Two photolithographically defined masters — one for the flow layer, one for



(a) Continuous-flow. **(b)** EWOD. **(c)** Two-phase slug flow.

Figure 2.1: The three dominant microfluidic biochip substrate families. (a) A continuous-flow chip with $\sim$1,000 individually addressable picolitre chambers and 3,574 microvalves, organized as a 25×40 array addressed via row and column multiplexors; reproduced from Thorsen *et al.* [41]. (b) The PurpleDrop EWOD device used by the Puddle runtime: aqueous droplets are moved across a 2D electrode grid (centre); reproduced from Willsey *et al.* [45]. (c) A T-junction droplet-formation geometry — the canonical slug-flow generator — with the dispersed aqueous phase (W_d) entering the continuous oil phase (W_c) perpendicularly, producing an aqueous slug (L) downstream; reproduced from Gu *et al.* [18].

the control layer — are replicated in PDMS, orthogonally aligned, plasma-bonded to glass, and cured to yield a sealed three-layer device with fluid channels separated from control channels by a thin PDMS membrane.

A *Quake valve* forms wherever a control line crosses a flow channel: pressurizing the control line deflects the membrane into the flow channel, occluding it; releasing the pressure reopens it. Actuation is fast (ms-scale) and repeatable. A *peristaltic pump* is formed by actuating three such valves along a short arc of flow channel in a fixed phase pattern, displacing a reproducible volume quantum γ per cycle — the unit of discretization for the entire architecture.

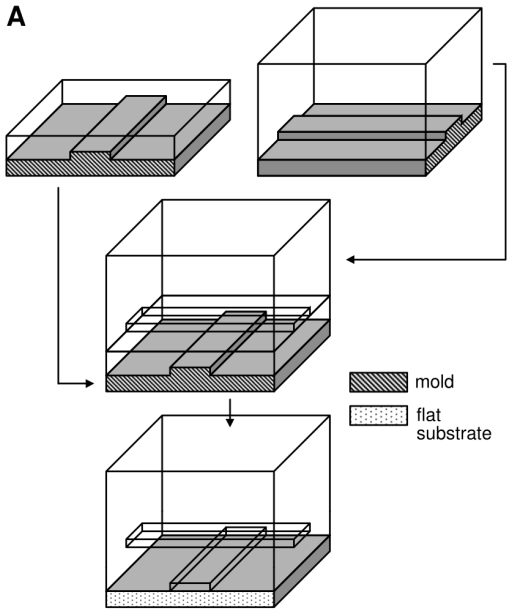


Figure 2.2: Multilayer soft-lithography fabrication of a Quake valve, after Unger *et al.* [42]. Two PDMS layers are separately cast against photoresist molds: the upper (control) layer carries pneumatic actuation lines; the lower (flow) layer carries the fluid channels. The layers are bonded with their channels orthogonal, separated by a thin PDMS membrane. A valve forms at every channel crossing: pressurising the control line above deflects the membrane downward into the rounded flow channel, occluding it. Reproduced from Unger *et al.* [42].

Two further results are load-bearing in later chapters. Goulpeau *et al.* [15] characterize the peristaltic pump in detail, establishing per-cycle displacement reproducible to within $\sim$ 1% in the linear regime,

and a per-chip ratio $\gamma_{\text{actual}}/\gamma$ ranging from 0.68 to 1.33 across fabrication tolerances. Gervais *et al.* [13] characterize flow-induced channel deformation, showing it is linear in the local pressure and inversely cubic in the elastomer-slab thickness above the channel.

2.3 Two-Phase Slug Flow Hydrodynamics

When an aqueous phase is introduced into an oil-wetted channel containing an immiscible carrier oil, it forms discrete slugs that the oil flow transports along the channel. Four phenomena underpin the HyDRA regime.

Capillary-dominated shape. At the operating geometry ($W \times H = 100 \times 30$ μm) and nominal velocity $v_{\text{nom}} = 10$ mm/s, the dimensionless groups are

$$Re = \rho v H/\mu_o \approx 0.39, \quad Ca = \mu_o v/\sigma = 7.3 \times 10^{-4}, \quad We = \rho v^2 H/\sigma \approx 2.8 \times 10^{-4}, \tag{2.1}$$

with μ_o the carrier-oil viscosity. Both $Re \ll 1$ (Stokes regime) and $We \ll 1$ (capillary-dominant) hold comfortably: slug-cap geometry is set by interfacial tension and channel geometry alone, independent of pump rate. This is the regime characterized exhaustively by Gilet and van Loo [14]; the consistent slug geometry across actuation speeds underlies much of the stability argument that follows.

Bretherton lubrication film. An advancing slug entrains a thin oil film of approximate thickness [7]

$$h_\infty \;\approx\; 1.34\,\frac{H}{2}\,Ca^{2/3} \;\approx\; 163 \text{ nm} \tag{2.2}$$

between itself and each wall. The Bretherton expression and the velocity-excess formula below are originally derived for circular tubes; in HyDRA's rectangular channel, aspect-ratio and corner-gutter corrections modify both quantities by up to $\sim 50\%$. The architecture is sized against bounds that absorb this correction (Chapter 6). Three architectural consequences follow: (i) the slug never contacts the wall during transport, so wall-adsorption contamination is minimized; (ii) the slug advances slightly faster than the mean carrier velocity, by a fraction $\alpha \approx 4h_\infty/H \approx 0.022$ [5] — a small per-stroke positional drift that the architecture's PARK re-registration zeroes at every checkpoint (Chapter 9); (iii) the wall film can drain during parking, raising the stability question addressed in Chapter 6.

Surfactant stabilization and controlled coalescence. Bare water–oil interfaces coalesce within microseconds under Laplace-pressure drainage; an architecture that preserves slug identity requires stabilization. The standard approach adds a PFPE-PEG diblock surfactant (Fluosurf-C, or equivalent) at sub-CMC concentration. The surfactant reduces interfacial tension — and hence the coalescence driving pressure — while sterically stabilizing the intervening oil film through brush–brush repulsion.

The same mechanism enables *controlled* coalescence at parked valves, provided coverage is kept low. Mazutis and Griffiths [28] show that low-coverage droplets fuse in milliseconds to seconds while high-coverage ($\gtrsim 90\%$) droplets remain kinetically stable for hours. HyDRA operates at 4–14% of saturation (Chapter 6).

Slug–valve interactions. Gilet and van Loo [14] classify slug–valve interactions into four regimes — breakup, splitting, gutter-directed pass-through, and trapped-film parking (**TF**) — as functions of capillary number and geometry. HyDRA's parking procedure (Chapter 9) operates in the TF regime at low velocity, pressing the slug gently against the valve face without splitting or gutter deformation.

These four phenomena fix the substrate vocabulary used throughout the rest of the thesis. Chapter 3 surveys prior attempts to build programmable abstractions over related substrates, and locates the gap that HyDRA fills.

Chapter 3

Literature Survey and Gap Analysis

This chapter surveys prior programmable-microfluidic systems, locates the gap that HyDRA fills, and distills the research questions answered by the rest of the thesis.

The microfluidic-biochip field traces to the application of soft-lithography to fluid handling in the 1990s. Whitesides [44] surveys the formative period and identifies multilayer PDMS with pneumatic actuation as the substrate that achieved the broadest reproducibility and the widest device-complexity range. Within a decade of Unger *et al.*'s introduction of Quake-style pneumatic valves [42], single-laboratory devices with thousands of addressable valves were demonstrated [41, 29]. Against this fabrication maturity, the programmability literature splits into two strata: design-automation tools that schedule low-level valve and electrode actuation (Section 3.1), and high-level programming abstractions that present a substrate-independent or substrate-specific programming model (Section 3.2). HyDRA's contribution sits at the second stratum.

3.1 Design Automation for Microfluidic Biochips

The continuous-flow design-automation literature has matured around two complementary problem classes. The first is *synthesis*: given an assay specification, derive a chip layout (channel topology, valve placement, pump arrangement) that supports the assay. The second is *scheduling*: given a fabricated chip and an assay, generate the time-indexed valve actuation sequence that executes the assay correctly and efficiently. Chakrabarty *et al.* [8] survey the synthesis side; Minhass *et al.* [30] provide a representative scheduling formulation, casting it as an architectural-synthesis problem analogous to high-level synthesis in digital VLSI.

These tools formalize substantial substrate-level knowledge — valve timing, fluid-routing constraints, pump contention — but operate *below* the abstraction level of interest to the present work. The output of such a tool is a flat sequence of valve actuations, equivalent to the assembly code emitted by a high-level-synthesis tool from RTL. The user's input is the assay graph, not a programmable instruction set; the output is a chip-specific actuation stream, not a compiled program in a substrate-independent abstraction. Reusing the actuation stream across chip layouts requires re-running synthesis. HyDRA's contribution is, by analogy, to the level above: an instruction-set abstraction that the design-automation layer can target, with compile-time guarantees that survive arbitrary lowering choices.

3.2 High-Level Programming Abstractions

Six prior systems have proposed high-level programming abstractions for microfluidic biochips. Each makes a defensible design trade along the three axes the field has historically struggled to satisfy simultaneously: (i) substrate independence, (ii) static substrate-level safety guarantees, and (iii) ahead-of-time verifiability of compiled programs.

3.2.1 BioCoder: A Substrate-Agnostic DSL

BioCoder [4] is a high-level language for specifying biological protocols at the level of reagent quantities, steps, timings, and actions (*mix*, *centrifuge*, *incubate*). Its design value is substrate-generality: a single description can drive a continuous-flow chip, an EWOD device, a robotic pipettor, or a human operator. The design cost is the converse: BioCoder does not formalize substrate-level state (open valves, occupied registers, resident volumes), so substrate-level correctness properties — "this split is impossible: the source slug is smaller than the requested aliquot," or "this operation leaves the chip in an unrecoverable state" — cannot be stated at the language level, much less checked. BioCoder is a protocol-description language, not an instruction set. It sits at a different position in the design space than HyDRA: complementary rather than competitive, since a BioCoder front end could in principle target a HyDRA back end.

3.2.2 BioStream and AquaCore: Continuous-Flow Architectures

BioStream [40] and AquaCore [3] are the two canonical proposals for continuous-flow programmable architectures. BioStream models the chip as a pipelined dataflow system in which reagents progress through mixing and splitting stages along fluid paths. AquaCore proposes a specific chip with processing elements, a register file of reaction chambers, and an on-chip interconnect, explicitly drawing the analogy with general-purpose processor architecture: reaction chambers as registers, on-chip interconnect as memory bus.

Both commit to a continuous-state substrate. The abstract state is a concentration field over a network of channels; the operational semantics describe how this field evolves under valve and pump actions. Within this abstraction, both systems support a programming model that can describe realistic biochemical workflows, and both have been demonstrated in simulation. Neither admits integer semantics: there is no notion of an aqueous body of discrete volume, no notion of a pump quantum, and consequently no static finite-state characterization of executions. Correctness arguments necessarily appeal to dispersion-aware fluid models, and the finite-integer reasoning that supports HyDRA's static guarantees is unavailable.

Both also inherit a structural composition challenge from continuous-flow chips: the dispersion of one operation propagates into the input of the next, unavoidably absent active barriers between operations. Slug flow eliminates this class of issues by construction: each slug is a discrete parcel separated from its neighbors by carrier oil, with no convective mixing between parcels.

3.2.3 BioScript: Typed Chemistry on EWOD

BioScript [31] is a statically-typed programming language for biochemical protocols on EWOD devices. Its central innovation is a type system that tracks chemical identity and safety properties of reagents, preventing protocols that mix reagents whose combination is known to be hazardous. The contribution is in the *chemical* dimension: substrate-level reasoning — which droplet occupies which grid cell, which routes are committed, which electrodes are energized — is deferred to the runtime.

This makes BioScript complementary to HyDRA rather than competitive: the type-system machinery is orthogonal to substrate-level verification. A future system could compose HyDRA's substrate invariants with BioScript-style chemistry types, giving the programmer a unified compile-time safety envelope spanning both the physical substrate and the chemistry. The chemistry-type annotations flow through SSA naming as auxiliary data and do not affect substrate-level lowering or invariants (Chapter 14).

3.2.4 Puddle: Runtime Error Correction on EWOD

Puddle [45] provides an EWOD programming abstraction centered on runtime detection and correction of droplet-movement failures. The system addresses a real concern: EWOD electrodes have nonzero defect rates, droplet motion can fail due to contamination, and electrode actuation can have non-deterministic outcomes near fabrication tolerances. Puddle's runtime monitors droplet positions during execution and reroutes around failed cells.

Puddle's guarantees are runtime, not compile-time. A program may fail at execution despite passing whatever static checks the compiler performs, and the runtime is responsible for detecting the failure and either correcting or aborting. HyDRA inverts this: every safety-relevant property is decided ahead of time, and any program that passes verification is guaranteed to execute without the covered failure modes (Chapter 8). The two systems take opposite stances on the static-vs-dynamic correctness trade, and the choice is grounded in a substrate-level observation: EWOD electrode failures are stochastic and substrate-bound, while the slug-flow failure modes HyDRA covers (valve safety, register overflow, slug separation) are entirely positional and so admit compile-time decision.

3.2.5 BioWare-CFP: Modular Interoperability

BioWare-CFP [39] is a modular cyber-fluidic platform in which disparate microfluidic components — pumps, mixers, analytical instruments — interoperate through a standardized interface. The architectural contribution is module-level: a single high-level assay description drives a chain of multivendor components.

BioWare-CFP does not prescribe an instruction set in the HyDRA sense; its design contribution is at a different layer (interoperability across heterogeneous modules). It is precedent rather than competitor: module-level interoperability is achievable with current technology, validating the architectural assumption that external modules can be attached to HyDRA's tile ports and driven by HyDRA's compiler without bespoke plumbing per module.

3.2.6 Summary and Comparative Position

Table 3.1 classifies the prior systems along three dimensions: substrate family (CF, EWOD, SF), abstract state character (continuous, discrete), and nature of static safety checking.

Table 3.1: Prior programmable-microfluidic systems compared to HyDRA. Substrate: CF continuous-flow, EWOD electrowetting-on-dielectric, SF slug-flow. State: C continuous, D discrete.

System	Substrate	State	Static check
BioCoder	various	—	—
BioStream	CF	C	—
AquaCore	CF	C	—
BioScript	EWOD	D	type safety
Puddle	EWOD	D	runtime only
BioWare-CFP	various	—	—
HyDRA	SF	D	substrate

No prior system combines a slug-flow substrate, an SSA-disciplined ISA, a finite integer state model, and asynchronous module contracts. This is HyDRA's specific position in the design space.

The pattern is familiar from general-purpose computing. Verified-compiler efforts such as CompCert [27] demonstrate that formal correctness guarantees are achievable when (i) the source language has a well-defined small-step operational semantics over a finite-representable state, (ii) the lowering to machine instructions is structured in compositional layers, and (iii) each layer's correctness is proved relative to the next layer's semantics. Each of HyDRA's three architectural contributions — the discrete state space established by the V/T/P principles (Chapter 5), the layered specification across microarchitecture, ISA, and formal model (Chapters 9–8), and the SSA discipline that bounds the live-identifier set (Chapter 10) — transposes one of these structural choices to the microfluidic substrate. The proof-mechanization step that completes the CompCert program is identified as future work (Chapter 14).

3.3 Gap Analysis

Synthesizing across the survey, no prior system simultaneously provides:

- a **slug-flow substrate** with discrete aqueous identity and one-dimensional topology, free of the dispersion coupling that limits continuous-flow architectures;
- a **single-static-assignment instruction set** in which every aqueous body has a stable compiler-tracked identifier whose provenance, lifetime, and consumption are statically analyzable;
- a **finite, integer-valued state model** in which every safety-relevant property reduces to a Boolean combination of integer linear inequalities, supporting ahead-of-time verification at cost polynomial in program size and live-slug count for bounded branch depth; and
- **native asynchronous module contracts** whose soundness is established by a formal invariant on the pending-future set, enabling pipeline parallelism on a serial-bus architecture.

These four together yield a compositional discipline: any sub-program that passes verification in isolation remains valid when embedded in a larger program, by virtue of all invariants holding at every ISA boundary. A system filling this combined gap offers two distinct classes of advantage. From the user's perspective, programs are written against a well-defined instruction set rather than against specific valve layouts, and programs that compile are guaranteed to execute correctly without the runtime correction infrastructure of EWOD systems or the dispersion-aware modeling of continuous-flow architectures. From the architectural perspective, the substrate admits the kinds of formal guarantees that the verified-compiler literature has established for general-purpose computing (Chapter 8).

3.4 Research Questions

The thesis answers five research questions, summarized in Table 3.2.

Table 3.2: Research questions answered by the thesis.

	Topic	Question	Answered in
RQ1	Physical realizability	Does a set of architectural primitives — pump quantum γ, register geometry, valve topology, operating fluid system — exist such that all volumes are integer multiples of a fixed unit, all positions are integer cell indices, and all operations are finite compositions of integer advances?	Chs. 5–7
RQ2	Joint operating window	Does a point in parameter space exist where both wall-film stability during transport and on-demand coalescence at parked valves hold simultaneously, given that the same surfactant governs both?	Ch. 6
RQ3	ISA expressiveness	Can an SSA-disciplined slug-functional ISA represent the class of realistic biochemical protocols within the architecture's resource envelope?	Chs. 10, 11
RQ4	Tractable verification	Are safety-relevant properties of HyDRA programs decidable ahead of time in polynomial resources?	Ch. 8
RQ5	Asynchronous parallelism	Can asynchronous module contracts recover pipeline parallelism on a serial-bus architecture, and at what throughput benefit?	Ch. 10

With the gap and research questions identified, Chapter 4 states the design objectives concretely, and the design chapters that follow answer each RQ in turn.

Chapter 4

Problem Statement and Objectives

4.1 Problem Statement

> *Design a programmable microfluidic architecture that exposes aqueous-slug handling as a discrete, integer-valued instruction set, whose every safety-relevant property is decidable ahead of time over a finite integer state space, and which is realizable in a standard multilayer-PDMS fabrication platform under a standard fluorocarbon-oil / PFPE-PEG surfactant / Aquapel-treated PDMS fluid system.*

The problem has four coupled faces — physical, microarchitectural, architectural, and formal — and the design task is to find a simultaneous fixed point in which all four co-satisfy. A clean ISA with no physical substrate is not a solution; a realizable substrate whose state is continuous does not admit integer-valued semantics. The chapters that follow develop the architecture as a joint solution.

4.2 Design Objectives

The research questions of Section 3.4 determine eight concrete design objectives. Table 4.1 lists them together with the chapters in which each is addressed and evaluated.

Table 4.1: HyDRA design objectives.

	Objective	Short description	Evaluated in
O1	Physical realizability	Every architectural element realizable in standard Quake-valve PDMS without novel fabrication	Ch. 7
O2	Integer-valued semantics	Volumes as integer multiples of γ; positions as integer cell indices; no continuous quantities above fluid physics	Ch. 5
O3	Fluid-system feasibility	Non-empty operating window for wall-film stability and on-demand coalescence, defensible from literature	Ch. 6
O4	Finite SSA-disciplined ISA	User-visible interface is a finite instruction set under single-static-assignment discipline	Ch. 10
O5	Static decidability	Every safety-relevant property decidable at compile time over the finite integer state space	Ch. 8
O6	Tractable verification	Verification cost bounded at under 6×10^5 integer comparisons for realistic workloads	Ch. 8
O7	Asynchronous module integration	External modules attach via ISA-level contracts; long-running modules support on-chip parallelism	Ch. 10
O8	Target-class coverage	PCR, immunoassay, serial dilution, protein crystallization, and glucose assay all expressible within $N = 10$ tiles, $V_{\text{reg}} = 416$ cells	Ch. 11

The objectives are not independently achievable. Advancing O2 at the expense of O3, for instance, is not

a valid trade-off: integer semantics matter only insofar as the underlying fluid system actually supports the regime they assume. The same coupling holds across pairs and triples of objectives, which is why the solution is presented as a single layered specification rather than as eight independently-defended claims.

4.3 Non-Objectives

The thesis does *not* claim:

- **Continuous-flow separations** (e.g. electrophoresis, chromatography), which require spatial concentration gradients incompatible with discrete slugs.
- **Single-slug volumes exceeding $V_{\text{reg}}\gamma \approx 250$ nL.** Total on-chip volume up to $N \cdot V_{\text{reg}}\gamma \approx 2.5$ μL is in scope; preparative-scale biochemistry is not.
- **Unbounded loops.** `repeat` provides compile-time unrolling of a fixed count; loops whose bound depends on runtime data would require dynamic program generation and are excluded. Every legal HyDRA program is a bounded conditional DAG.
- **Fabrication, measurement, or machine-checked proofs** — a fabricated chip, a reference compiler implementation, measured assay execution, and a machine-checked correctness theorem are each identified as follow-up work in Chapter 14.

Chapter 5

Architectural Principles

Three architectural principles, imposed by the physics of the substrate, propagate upward into every abstraction layer and together form the bridge between the continuous-valued physical substrate and the discrete-valued ISA semantics of Chapter 10.

V — Volume quantization. All aqueous volumes are integer multiples of the pump quantum $\gamma = 0.6$ nL:

$$\forall\, s \in \Sigma :\ \mathrm{vol}(s) \in \gamma \cdot \mathbb{N}_{>0}. \tag{5.1}$$

T — Always-full sealed-loop hydraulic transmission. The channel network is filled with carrier oil at start-up and remains full throughout operation; a single pump stroke advances every on-path cross-section by exactly one cell:

$$\forall\, c \in \mathrm{ActivePath}(C) :\ \Delta x(c) = 1 \text{ cell per stroke}. \tag{5.2}$$

P — Single active flow path. At every instant, the valve configuration C partitions the channel network into one closed active path and a complementary set of off-path segments:

$$\mathrm{ActivePath}(C) \sqcup \mathrm{OffPath}(C) = \mathrm{Channels}, \quad \mathrm{ActivePath}(C) \text{ closed}. \tag{5.3}$$

The rest of the chapter establishes each principle: how the physical substrate supports it, what quantitative constraints it imposes on the architecture, and what must be audited to preserve it.

5.1 Volume Quantization

The pump quantum γ is the volume displaced by one cycle of the peristaltic pump. For a Quake-style three-valve peristaltic pump [42, 15] on a rectangular channel of width W and height H, γ is set by the valve membrane length ℓ_v:

$$\gamma = W \cdot H \cdot \ell_v, \tag{5.4}$$

where ℓ_v is the minimum channel segment a single Quake valve can isolate. For HyDRA's nominal geometry ($W = 100$ μm, $H = 30$ μm, $\ell_v = 200$ μm), this gives $\gamma = 0.6$ nL.

5.1.1 Stroke-Count Abstraction

Every volume is an integer stroke count; every distance is an integer cell count, where one cell is ℓ_v of channel length. A positive-displacement peristaltic pump produces sub-1% stroke-to-stroke variation

in the linear regime [15], advancing the fluid column by one cell per cycle of physical length $\ell_{\text{actual}} = \gamma_{\text{actual}}/(WH)$.

This abstraction cleanly separates two concerns. *Routing safety* — register overflow, slug collision, volume conservation — is stated over stroke counts and verified at compile time over integers. *Assay accuracy* — absolute concentrations and mixing ratios — depends on the per-chip value γ_{actual} and is a standard instrument-calibration concern. The separation holds because the hardware maintains it: uniform cross-section ($W \times H$ throughout) gives uniform advance; register oversizing against worst-case γ_{actual} absorbs capacity slack; periodic re-registration (Chapter 9) zeros accumulated positional drift.

5.1.2 Pump Error

Two distinct sources of pump-volume error must be managed: a systematic offset, absorbed once at commissioning and constant thereafter, and a stochastic per-stroke fluctuation that must be repeatedly zeroed. Conflating them leads to incorrect error accounting.

Systematic offset. The per-chip constant $\gamma_{\text{actual}} - \gamma$ reflects fabrication tolerances — channel dimensions, membrane thickness, mold quality, actuation-pressure tuning. Goulpeau *et al.* [15] measured the displacement ratio V_0/V_v across ten pumps of multiple geometries and observed values in $[0.68, 1.33]$, a worst-case $\pm 33\%$ deviation from nominal before per-pump tuning. After pressure optimization at the linear operating point, the spread tightens to $\sim \pm 20\%$.

This offset is absorbed at chip commissioning by a single per-chip scalar

$$s = \gamma/\gamma_{\text{actual}}, \tag{5.5}$$

applied by the compiler at cell-to-stroke translation: a logical advance of n cells is realized as $n_{\text{phys}} = \lceil n \cdot s \rceil$ physical strokes. The rounding absorbs at most one cell per calibration boundary and propagates no further. Register sizing is set in cells, independent of the per-chip γ_{actual}: $V_{\text{reg}} = 416$ cells gives a nominal capacity of 416×0.6 nL $= 249.6$ nL at the nominal γ. The physical volume held in a register varies with γ_{actual} (from 170 nL at $\gamma_{\text{actual}} = 0.68\gamma$ to 332 nL at $\gamma_{\text{actual}} = 1.33\gamma$); the calibration scalar maps the user-specified target volume onto an integer cell count, ensuring a single slug never exceeds V_{reg} cells regardless of γ_{actual}.

Stochastic deviation. The per-stroke fluctuation around γ_{actual} accumulates *within* an operation. Define ε_k such that the actual displacement on stroke k is $\gamma_{\text{actual}}(1 + \varepsilon_k)$. Goulpeau *et al.* [15] bound this to sub-one-percent:

$$|\varepsilon_k| \leq \varepsilon_{\max} = 0.01. \tag{5.6}$$

The cumulative position-uncertainty budget combining ε with the Bretherton slug drift $|\alpha| \leq 0.03$ (Chapter 6) is developed in Chapter 9, where the PARK re-registration procedure that zeros it is also specified.

5.2 Always-Full Sealed-Loop Hydraulic Transmission

One pump stroke produces equal volume displacement at every cross-section of the active circuit — the principle that makes a single peristaltic pump sufficient to drive all fluid handling. It rests on three conditions: no compressible voids in the loop (Section 5.2.1); a valve configuration that partitions the network into one dominant flow path and high-impedance off-path segments (Section 5.2.2); and transient elastic deformation within each stroke small enough that advance is effectively simultaneous (Section 5.2.3).

5.2.1 Always-Full Condition

The loop is primed with carrier oil before first use and is never partially drained; no free surfaces, gas phases, or voids exist during operation. Any void would introduce compressibility and break the 1:1

correspondence between pump displacement and distant-point advance. The correspondence holds at cell resolution provided that wall compliance and fluid compressibility together stay below one pump quantum over the full loop (Section 5.2.3). Operations that add or remove aqueous material must simultaneously displace an equal volume of carrier oil; the I/O zone (Chapter 7, Section 7.6.3) provides this off-chip exchange via gated stub channels.

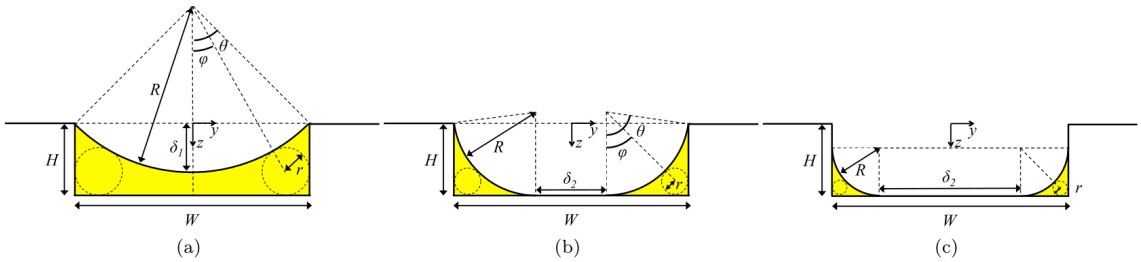
5.2.2 Off-Path Stability

Principle T requires that off-path segments contribute negligibly to pump displacement: closed valves must not bleed enough to perturb on-path advance. The architectural question is whether bypassed registers — the dominant off-path component in any non-trivial configuration — meet this requirement.

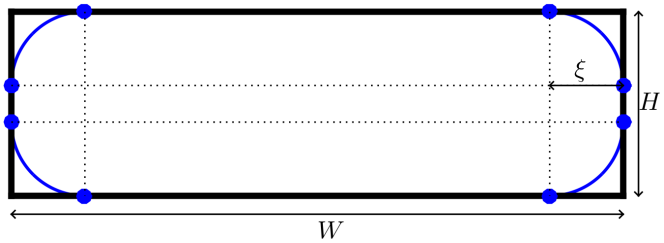
The pressure driving cross-valve leakage is the Hagen–Poiseuille drop over one cell:

$$\Delta P_{\text{cell}} = \frac{12\mu_o v \ell_v}{H^2} \approx 33 \text{ Pa} \qquad (\text{at } v_{\text{nom}} = 10 \text{ mm/s}). \tag{5.7}$$

This drives leakage through the sequence $V_1 \rightarrow$ serpentine $\rightarrow V_2$ of a bypassed register. Quake valves do not perfectly seal rectangular channels: when the flat PDMS membrane deflects against the floor, its curved profile leaves two longitudinal gutters along the bottom side-wall corners [14]. The closed-valve resistance is $R_{\text{valve}} = \beta_{\text{closed}}\mu_o\ell_v/H^4$ with $\beta_{\text{closed}} \approx 725$ from Gilet and van Loo (their Fig. 12 at the HyDRA geometry), about 160× the open-channel coefficient.



(a) Three regimes of PDMS membrane deformation under increasing pneumatic pressure P_v. (a) Suspended arc; (b) tangent to channel bottom over distance δ_2; (c) tangent to bottom and side walls. The shaded corner regions remain accessible to carrier oil in all three regimes — closed valves leak through these gutters by construction.



(b) Cross-section of a parked slug inside a rectangular channel of width W and height H. The aqueous slug (blue) is pinned by interfacial tension to the channel's four side-wall corners; the four corner gutters of radius ξ remain available for carrier-oil bypass.

Figure 5.1: The two corner-gutter geometries that maintain HyDRA's always-full condition under closed valves and parked slugs. (a) Closed Quake valve: the deflected PDMS membrane leaves two corner gutters open along the bottom side walls. (b) Parked slug: the aqueous body is pinned at the four channel corners, leaving four corner gutters of radius $\xi \approx 12$ μm available for oil bypass. Both panels reproduced from Gilet and van Loo [14].

Empty bypassed register. The leakage-path resistance $R_{\text{leak}} = 2R_{\text{valve}} + R_{\text{serp}} \approx 9 \times 10^{14}$ Pa·s/m^3 compares against the bypass cell's $R_{\text{cell}} \approx 1.1 \times 10^{12}$ Pa·s/m^3. The leakage fraction is $\sim 0.12\%$ of main flow — roughly one cell of displacement inside the register per full loop transit. Since there is no slug to displace, this is architecturally inert.

Occupied bypassed register. When a slug is parked inside a bypassed register, it enters Gilet's regime TF [14]: oil bypasses the trapped slug through its four corner gutters (two from the deflected valve membrane, two from the slug's channel-corner menisci). The additional resistance is

$$R_d \approx 2 \times 10^5 \left(\frac{L_d}{W} - 1 \right) \frac{\mu_o}{H^3}, \tag{5.8}$$

which for a fully-filled register slug ($L_d = 83$ mm, $L_d/W = 830$) evaluates to $R_d \approx 8 \times 10^{18}$ Pa·s/m^3 — four orders of magnitude above the empty-register path. Leakage drops to $\lesssim 2 \times 10^{-7}$ of main flow, far below one cell of displacement over the lifetime of any program. Bypassed storage is mechanically stable *by virtue of TF-regime resistance*.

The gutter bypass is not a defect but functionally essential: the PARK procedure of Chapter 9 relies on oil flowing through corner gutters past a parked slug to establish valve contact without splitting the slug.

5.2.3 Compliance

In a closed-loop positive-displacement system, mass conservation guarantees that γ_{actual} of fluid passes every cross-section per cycle regardless of compliance; compliance does not introduce *positional* error. What it does affect is the *transient* behavior within each stroke: the Poiseuille pressure gradient compresses fluid and deflects PDMS walls on the high-pressure side. The hydraulic-transmission assumption — that all on-path fluid advances approximately simultaneously within each stroke — requires this transient deformation to be small relative to γ.

Table 5.1: Compliance parameters.

Symbol	Value	Description
A	$WH = 3 \times 10^{-9}$ m^2	Channel cross-section
L	$L(C)$	Active path length
K	$\rho c^2 \approx 0.77$ GPa	HFE-7500 bulk modulus
h_{slab}	4 mm	PDMS slab thickness above channel
E_{PDMS}	1.5 MPa	Young's modulus, Sylgard 184 (10:1)
α_p	≈ 0.014	Plate-bending coefficient [13]

Fluid compressibility. Compression of the fluid column over the active path under internal pressure P:

$$\delta V_{\text{fluid}} = \frac{P}{K} AL. \tag{5.9}$$

Wall compliance. For a rectangular channel in a thick elastomer slab ($h_{\text{slab}} \gg W$), the ceiling deflection follows classical clamped-plate bending [13]:

$$\delta V_{\text{wall}} \approx \tfrac{2}{3} \alpha_p \frac{P W^4}{E_{\text{PDMS}}\, h_{\text{slab}}^3} WL. \tag{5.10}$$

At $W = 100$ μm and $h_{\text{slab}} = 4$ mm, $\delta V_{\text{wall}}/\delta V_{\text{fluid}} \lesssim 0.03\%$ across the operating pressure range — negligible. Wall compliance is omitted from the summary table below.

Operating-pressure scaling. The driving pressure is the Hagen–Poiseuille drop

$$\Delta P_{\text{flow}} = \frac{12 \mu_o v L}{H^2}.$$

Both ΔP_{flow} and the compressible volume AL scale with L, so $\delta V_{\text{fluid}} \propto L^2$. Each XC tile adds about 83 mm to the path; at the nominal velocity $v_{\text{nom}} = 10$ mm/s:

We bound compressibility-induced drift at $\delta V_{\text{fluid}}/\gamma < 10\%$ — an order of magnitude above the per-stroke

Table 5.2: Compressibility at nominal velocity, as fraction of γ.

Configuration	L (mm)	ΔP_{flow} (kPa)	δV_{fluid} (nL)	$\delta V/\gamma$
All-Bypass	124	20.5	0.010	1.7%
1 register XC	207	34.3	0.028	4.6%
2 registers XC	290	48.1	0.055	9.1%

pump deviation $\varepsilon_{\max} = 1\%$ — so that compliance remains a sub-dominant uncertainty source relative to pump variation. This bounds the simultaneous XC count:

$$\boxed{n_{\text{XC}} \;=\; \sum_{i=0}^{9} \mathbb{1}[\text{tile}_i = \text{XC}] \;\le\; 2.} \tag{5.11}$$

This is a compile-time invariant enforced by the scheduler (Chapter 8). At $n_{\text{XC}} \le 2$, the hydraulic transmission is quasi-rigid: fluid advance is effectively simultaneous across the active path within each stroke, and mass conservation guarantees the exact per-stroke displacement γ_{actual} at every cross-section.

Temporal separation. The compliance RC timescale $\tau_{\text{RC}} = R_{\text{hyd}} C_{\text{hyd}} = 12\mu_o L^2/(KH^2)$ at the worst-case $L = 290$ mm evaluates to $\tau_{\text{RC}} \approx 1.8$ ms — an order of magnitude below the 20 ms stroke period. Compliance dynamics are fully separated from pump dynamics; any residual transient dissipates well inside a single stroke cycle.

5.3 Single Active Flow Path

Principle P requires that the valve configuration always realize *exactly one* closed active path. Two flow paths simultaneously open would split pump displacement between them according to transient hydraulic resistance rather than ISA intent, breaking the deterministic correspondence between strokes and abstract operations.

The principle is enforced microarchitecturally by *break-before-make* discipline: any tile reconfiguration with the pump active proceeds through valve transitions that close one configuration before opening the next, never both. The transient single-valve modes used for re-registration (AC and RP, Chapter 7) are entered only with the pump idle, so the active-path partition is well-defined whenever fluid is in motion.

A consequence is that the abstract-level tile-mode vector $C \in \{\mathsf{B}, \mathsf{X}\}^N$ — where B and X abbreviate the standard tile modes BP and XC of Chapter 7 when used as components of the abstract state — is sufficient to determine the active path. The transient single-valve modes and the module-mode MD (which holds the pump) do not appear at the abstract level: the formal state (Σ, C, ν, F) of Chapter 8 is inspected only between completed ISA operations, at which point the chip is always in a static, well-defined mode.

Chapter 6

Two-Phase Operating Regime

The architectural model of Chapter 5 treats slugs as integer-valued objects on a one-dimensional cell grid, and the physical layout of Chapter 7 translates that grid into channels, valves, and ports. Between these two views sits the fluid regime: the choice of carrier oil, surfactant, and surfactant concentration, plus the dimensionless flow regime in which the slugs are operated. The regime is what makes the abstraction physically honest. Pump strokes can map cleanly to cell advances only if the slug body fills the channel without leaks, the lubricating wall film is mechanically stable for as long as a slug is parked, and on-demand coalescence at a chosen valve is achievable in a bounded time. This chapter establishes that an operating window exists in which all three conditions hold simultaneously, and pins HyDRA's parameters to a representative point inside it.

6.1 Three Physical Requirements

R1 — Wall-film stability. Carrier oil must persist as a thin film between every slug and every channel wall throughout transport and parking. Loss of the wall film exposes the aqueous slug to the PDMS surface, breaking the encapsulation invariant of Chapter 5 and admitting both surface adhesion and uncontrolled cross-channel transport. Formally, the steric component of the disjoining pressure profile must dominate the van der Waals component at some equilibrium thickness $h_{\min} > 0$, with the net peak exceeding the corner-suction driving pressure: $\Pi_{\text{steric}}(h_{\min}) - |\Pi_{\text{vdW}}(h_{\min})| \geq \Delta P_{\text{corner}}$.

R2 — Coalescence on demand. Two slugs parked against opposite faces of a closed valve must coalesce within a bounded time window after the valve opens. The MERGE operation of Chapter 10 compiles directly into this physical event; without it, the architecture cannot construct the multi-reagent slugs required by every assay class identified in Chapter 3. Quantitatively, coalescence must complete reliably under the geometric and thermodynamic conditions imposed by valve retraction and the bus pressure budget; the architecture schedules a ~ 10 s WAIT after every MERGE as a conservative budget against the millisecond-to-second timescale predicted by the experimental literature (Section 6.5.3).

R3 — Stable two-phase slug flow. During CIRC, slugs must traverse the channel network without break-up, satellite generation, or stratification. Stable two-phase flow in rectangular microchannels persists up to $Ca \sim 10^{-2}$ [14]; as a conservative design margin we require $Ca < 10^{-3}$ at every operating velocity.

R1 and R2 pull in opposite directions: increasing surfactant coverage increases wall-film stability but suppresses coalescence; decreasing it does the reverse. The architecture's central physical claim is that the asymmetry between the wall-film and slug-slug Hamaker constants opens a window in which both R1 and R2 hold, and that HyDRA's chosen fluid system sits inside that window with margin. The remainder of the chapter develops this claim quantitatively. R3 is satisfied trivially at every operating point and is checked along the way.

Scope and conditional validity. The quantitative results in Sections 6.5.2 and 6.5.3 are conditional on two material parameters that must be measured per device: the surfactant brush thickness L_0 and the adsorption isotherm relating bulk surfactant concentration to interfacial coverage Γ. Throughout this chapter we use $L_0 = 4$ nm as a representative value for ~ 10 kDa PFPE-PEG surfactants [2] and treat the grafting distance s (or equivalently $\Gamma/\Gamma_{\text{sat}}$) as the design variable. The R2 (coalescence) feasibility argument rests on three independent experimental anchors from the droplet-microfluidics literature [28, 34, 10], which together place coalescence in HyDRA's regime on the millisecond-to-second timescale.

6.2 Slug Drift

Chapter 2 fixes the dimensionless flow regime — $Re \approx 0.39$, $Ca = 7.3 \times 10^{-4}$, $We \approx 2.8 \times 10^{-4}$ at v_{nom} — and reports the Bretherton lubrication-film thickness $h_\infty \approx 163$ nm at this Ca [7]. R3 follows immediately: Ca is below the 10^{-3} design margin at v_{nom} and falls another decade at v_{prec}. The remaining architectural consequence to develop is slug drift — a small persistent excess of slug velocity over mean carrier velocity, produced by the Bretherton film together with the corner gutters, which accumulates over each pump stroke and is the physical reason the discrete model needs PARK re-registration.

The Bretherton film and the corner gutters together carry a fraction of the channel cross-section that is essentially stagnant relative to the moving slug body. The film is ~ 163 nm thin [7], and the four corner gutters in parallel have per-unit-length hydraulic resistance more than 5,000× that of the main channel (Ransohoff–Radke corner-flow scaling, $\beta \approx 94$) [46]. Mass conservation requires the slug body to advance *faster* than the mean carrier velocity to compensate for the volume that does not move with it. Define the drift fraction α such that

$$v_{\text{slug}} = v_{\text{carrier}}\,(1 + \alpha). \tag{6.1}$$

For circular tubes, the Aussillous–Quéré correlation [5] gives the relative film thickness:

$$\frac{h}{R} = \frac{1.34\,Ca^{2/3}}{1 + 3.35\,(3Ca)^{2/3}}. \tag{6.2}$$

Mass conservation in the thin-film limit ($h \ll R$) gives $\alpha \approx 2h/R$. At $Ca = 7.3 \times 10^{-4}$ this yields $h/R \approx 0.010$ and $\alpha \approx 0.021$. Rectangular-channel corrections (corner-gutter geometry, aspect-ratio-dependent film profile) modify the bare Aussillous–Quéré prediction by up to 50% [46]. Adopting a conservative bound for the discrete model,

$$\boxed{|\alpha| \le 0.03} \tag{6.3}$$

is the slug-drift contribution to the per-stroke advance ratio used throughout Chapter 5 and Chapter 9. Per stroke, the slug body advances $1 + \alpha$ cells instead of exactly 1. Over n strokes the accumulated positional drift is $n\alpha$ cells; over a full all-Bypass loop transit ($n \approx L_{\text{bus}} = 619$ strokes), worst-case drift is ~ 19 cells or ~ 4 mm. This drift is the physical motivation for the PARK re-registration primitive, which contacts a closed valve membrane to zero accumulated drift and is specified in Chapter 9.

6.3 Two-Phase Physics

Three physical effects in the channel wall and in the corner gutters together determine where the slug surface sits, what holds the wall film up, and what drives it down. We address them in turn: wetting (does oil spread, in principle, between slug and wall?), corner geometry (where does the slug interface go in a non-circular channel?), and disjoining pressure (what equilibrium does the wall film reach when it stops draining?).

6.3.1 Wetting

Whether oil spontaneously intercalates between an aqueous slug and a PDMS wall is determined by the spreading coefficient

$$S = \gamma_{SW} - \gamma_{SO} - \gamma_{OW}, \tag{6.4}$$

where γ_{SW}, γ_{SO}, γ_{OW} are the solid–water, solid–oil, and oil–water interfacial tensions respectively [11]. $S > 0$ is the thermodynamic precondition for an oil film between slug and wall; the opposite sign means the slug pins to the wall. For HyDRA's HFE-7500/water/Aquapel-PDMS combination, $S > 0$ is operationally confirmed in the slug-flow regime that Chapter 2 adopts. Aquapel treatment is what makes this hold robustly: untreated PDMS swells in fluorinated solvents and the wetting hierarchy can invert [26].

6.3.2 Corner Geometry

Slug interfaces cannot fill the sharp corners of a rectangular channel: the Laplace pressure required for interfacial curvature to follow the corner diverges as the corner radius vanishes. Instead, the interface adopts a rounded meniscus of radius ξ_b, leaving four longitudinal corner gutters open along the channel [46]:

$$\xi_b = \frac{H + W - \sqrt{(H+W)^2 - (4-\pi)HW}}{4-\pi} \approx 12\ \mu\text{m}. \tag{6.5}$$

The corner gutters serve two architectural roles. First, they carry oil past a parked slug when a closed valve is being pushed onto: pump strokes that exceed contact are absorbed by gutter bypass rather than by deforming the slug body, which is what allows the PARK procedure of Chapter 9 to terminate cleanly even in worst-case under-/over-delivery. Second, they sustain the corner-suction driving pressure that drains the parked-slug wall film toward equilibrium:

$$\Delta P_{\text{corner}} = \frac{\sigma}{\xi_b} \approx 1{,}417\ \text{Pa}. \tag{6.6}$$

This corner-suction pressure is the load that the steric barrier of Section 6.3.3 must support to keep the wall film stable. The wall-film equilibrium thickness $h_{\min}$ is set by the balance $\Pi_{\text{net}}(h_{\min}) = \Delta P_{\text{corner}}$.

6.3.3 Disjoining Pressure

The film between two interfaces separated by a thickness h exerts a disjoining pressure $\Pi(h)$ — the excess pressure relative to the bulk fluid that arises from molecular interactions across the film. For HyDRA's fluid system the electrical double-layer term is absent (fluorocarbon oils carry no free ions [22]), leaving two contributions:

$$\Pi(h) = \Pi_{\text{vdW}}(h) + \Pi_{\text{steric}}(h). \tag{6.7}$$

Van der Waals contribution. For a thin film of medium j between media i and k, the non-retarded vdW disjoining pressure follows the Hamaker form [22]:

$$\Pi_{\text{vdW}}(h) = -\frac{A_{ijk}}{6\pi h^3}. \tag{6.8}$$

For HyDRA's two relevant films $A_{ijk} > 0$, so the vdW contribution is always attractive (destabilizing) and grows as h^{-3}. The Hamaker constants A_{123} (PDMS–HFE-7500–water, the wall film) and A_{323}

(water–HFE-7500–water, the slug–slug film) are derived in Section 6.5.1.

Steric contribution. The PFPE-PEG surfactant adsorbs at the oil–water interface as an Alexander–de Gennes brush [2]: the PEG block extends into the aqueous phase to a thickness L_0, with chain endpoints separated by an interfacial grafting distance s. When two such brushes are compressed within $h < L_0$ (single-brush case, wall film) or $h < 2L_0$ (double-brush case, slug–slug film), the brush exerts a repulsive osmotic pressure:

$$\Pi_{\text{steric}}(h) = \frac{k_B T}{s^3}\left[\left(\frac{L_0}{h}\right)^{9/4} - \left(\frac{h}{L_0}\right)^{3/4}\right], \quad h < L_0. \tag{6.9}$$

The exponents are decisive: $|\Pi_{\text{vdW}}| \sim h^{-3}$ grows faster than $\Pi_{\text{steric}} \sim h^{-9/4}$ as $h \to 0$. Above some crossover thickness, the steric repulsion dominates and the film equilibrates at finite $h_{\min}$; below the crossover, vdW dominates and the film ruptures spontaneously — the fundamental coalescence mechanism. Whether the wall-film equilibrium thickness $h_{\min}$ exists at all depends on whether the peak of $\Pi_{\text{steric}} - |\Pi_{\text{vdW}}|$ exceeds the corner-suction load ΔP_{corner}. This is exactly the R1 stability condition; making it quantitative requires the Hamaker constants of Section 6.5.1.

6.4 Fluid Selection

The choice of carrier oil and surfactant determines the entire quantitative landscape of Sections 6.5.1–6.5.3. Both choices are constrained by R1, R2, R3 and by the fabrication context (PDMS multilayer soft lithography). The selections below are made narrow on purpose: each choice replaces a free dimension with a literature-anchored value, and the rest of the chapter inherits those values.

6.4.1 Carrier Oil

Selection criteria: water-immiscible, low viscosity (low Ca at the design velocity), minimal PDMS swelling [26], and compatibility with fluorinated surfactants. Two fluorinated carriers dominate the literature; we compare them in Table 6.1.

Table 6.1: Carrier-oil candidates and rationale.

Carrier	μ (mPa·s)	σ_{bare} (mN/m)	PDMS swelling	Disposition
HFE-7500 [1]	1.24	~ 40	Negligible [26]	Selected
FC-40	1.8	~ 52	Negligible	Eliminated: higher σ tightens R1 by raising ΔP_{corner}

HFE-7500 is selected. Its lower viscosity gives lower Ca at the design velocity, leaving more headroom in R3 and yielding a thinner Bretherton film. Crucially, its lower bare oil–water IFT (relative to FC-40) gives a smaller corner-suction ΔP_{corner}, which directly relaxes the steric-barrier requirement that R1 imposes on the surfactant. Both fluids are negligibly absorbed by PDMS over the timescales of HyDRA's operations.

6.4.2 Surfactant

Selection criteria: HFE-7500-soluble, fluorinated tail (compatible with the carrier), well-characterized steric mechanics (so L_0 has a meaningful literature anchor), and a sub-CMC operating regime accessible without exotic chemistry. We consider two candidates in Table 6.2.

Fluosurf-C is selected. Its CMC in HFE-7500 is approximately 0.2% w/w. Gilet & van Loo's baseline droplet-microfluidics system uses 0.5% w/w [14] — well above CMC, providing fully-saturated steric stabilization at every interface. HyDRA *deliberately operates sub-CMC* so that the interfacial coverage $\Gamma/\Gamma_{\text{sat}}$ sits well below the suppression threshold for passive coalescence ($\sim 90\%$ [28, 34]), admitting on-demand merge while keeping the wall-film steric barrier far above the corner-suction load. The R1 and

Table 6.2: Fluorinated surfactant candidates.

Surfactant	Structure	MW (kDa)	Disposition
Fluosurf-C [12]	PFPE–PEG diblock	7–13	Selected (Gilet baseline [14])
Pico-Surf 1	PFPE–PEG–PFPE triblock	~ 10	Eliminated: triblock geometry gives a stronger steric barrier, narrowing the R2 window

R2 requirements share a single operating variable, surfactant coverage, with R1 setting a lower bound and R2 an upper bound; the architecturally targeted band is 4–14% of saturation (Section 6.5.3).

6.5 Quantitative Analysis

The analysis in this section makes the R1 (wall-film stability) and R2 (coalescence on demand) requirements quantitative. The pivot is the asymmetry between two Hamaker constants: the wall film is stabilized by a small attractive vdW interaction across PDMS–oil–water, while the slug–slug film is destabilized by a substantially larger attractive vdW interaction across water–oil–water. The same surfactant brush, at the same coverage, supports the wall film while admitting slug–slug coalescence under capillary confinement.

6.5.1 Hamaker Constants

The Hamaker constant A_{ijk} governs the magnitude of the vdW disjoining pressure across a film of medium j separating media i and k (the middle subscript is always the film material [22]). We compute A_{ijk} via the Tabor–Winterton single-oscillator Lifshitz approximation [38],

$$A_{ijk} = T_0 + T_{\mathrm{UV}}, \tag{6.10}$$

separating the static (zero-frequency, dielectric) and ultraviolet (refractive-index) contributions. The material optical and dielectric properties used in the calculation are collected in Table 6.3.

Table 6.3: Optical and dielectric properties for the Hamaker calculation. Indices $1, 2, 3$ are reused throughout this section.

Index	Material	n	ε	Source
1	PDMS	1.410	2.7	Johnston *et al.* [23]
2	HFE-7500	1.290	5.8	3M datasheet [1]
3	Water	1.333	80	Standard [22]

Throughout the calculation, $k_B T = 4.1 \times 10^{-21}$ J at $T = 298$ K, and $h\nu_e = 2.0 \times 10^{-18}$ J for $\nu_e \approx 3 \times 10^{15}$ Hz [38].

Wall-film constant A_{123}. The asymmetric stack PDMS–HFE-7500–water gives:

$$T_0 = \frac{3k_B T}{4} \cdot \frac{(\varepsilon_1 - \varepsilon_2)(\varepsilon_3 - \varepsilon_2)}{(\varepsilon_1 + \varepsilon_2)(\varepsilon_3 + \varepsilon_2)} \approx -9.7 \times 10^{-22}\ \mathrm{J}, \tag{6.11}$$

$$T_{\mathrm{UV}} = \frac{3h\nu_e}{8\sqrt{2}} \cdot \frac{(n_1^2 - n_2^2)(n_3^2 - n_2^2)}{\sqrt{n_1^2 + n_2^2}\,\sqrt{n_3^2 + n_2^2}\,(\sqrt{n_1^2 + n_2^2} + \sqrt{n_3^2 + n_2^2})} \approx +1.45 \times 10^{-21}\ \mathrm{J}, \tag{6.12}$$

$$A_{123} \approx 4.8 \times 10^{-22}\ \mathrm{J} \quad \text{(net attractive — destabilizing)}. \tag{6.13}$$

The static term T_0 is negative because the dielectric constants of PDMS ($\varepsilon_1 = 2.7$) and HFE-7500 ($\varepsilon_2 = 5.8$) sit on the same side of the median and the asymmetric stack favors expansion of the oil film; the UV term, dominated by the refractive-index contrast between PDMS and HFE-7500, is positive and larger in magnitude. The result is a small net attractive A_{123}.

Slug–slug constant A_{323}. The symmetric stack water–HFE-7500–water guarantees $A_{323} > 0$ [38], with the static term dominating because of water's large ε:

$$T_0 = \frac{3k_BT}{4}\left(\frac{\varepsilon_3 - \varepsilon_2}{\varepsilon_3 + \varepsilon_2}\right)^2 \approx +2.30 \times 10^{-21}\ \text{J}, \tag{6.14}$$

$$T_{\text{UV}} \approx +5.3 \times 10^{-22}\ \text{J}, \tag{6.15}$$

$$A_{323} \approx 2.83 \times 10^{-21}\ \text{J} \quad \text{(always attractive)}. \tag{6.16}$$

The asymmetry. The crucial result is

$$\frac{A_{323}}{A_{123}} \approx 5.9. \tag{6.17}$$

The vdW attraction across the slug–slug film is approximately six times stronger than across the wall film. This is the physical foundation of the architecture's R1+R2 simultaneous satisfiability: the same steric brush, at the same surfactant coverage, supports the wall film (against a relatively small destabilizing vdW pressure) while admitting slug–slug coalescence under capillary confinement (where the destabilizing vdW pressure is six times larger and far more easily overcomes the brush). Without this asymmetry, R1 and R2 would be in direct conflict; with it, they coexist with margin.

6.5.2 Wall-Film Stability (R1)

The wall film is stable when the net disjoining pressure $\Pi_{\text{steric}}(h_{\min}) - |\Pi_{\text{vdW}}(h_{\min})|$ has a peak above the corner-suction load $\Delta P_{\text{corner}} \approx 1{,}417$ Pa at some equilibrium thickness $h_{\min}$. Substituting Equations 6.8 and 6.9, the equilibrium condition is

$$\frac{k_BT}{s^3}\left[\left(\frac{L_0}{h_{\min}}\right)^{9/4} - \left(\frac{h_{\min}}{L_0}\right)^{3/4}\right] = \frac{\sigma}{\xi_b} + \frac{A_{123}}{6\pi h_{\min}^3}. \tag{6.18}$$

R1 is satisfied when Equation 6.18 admits a solution with the peak of the left-hand side exceeding the right-hand side at the same $h_{\min}$. The grafting distance s is the design lever: smaller s means denser surfactant coverage and a stronger steric barrier. We numerically scan over s at $L_0 = 4$ nm and $\sigma = 17$ mN/m to locate the critical $s_{\max}$ above which the steric barrier no longer supports the corner-suction load.

Table 6.4: Numerical scan of the net disjoining-pressure peak as a function of grafting distance s, at $L_0 = 4$ nm and $\sigma = 17$ mN/m.

s (nm)	Peak net Π (Pa)	Stable against $\Delta P_{\text{corner}} = 1{,}417$ Pa?
10	$> 25{,}000$	Yes (margin $> 18\times$)
12	$> 12{,}000$	Yes (margin $> 8\times$)
14	$\sim 5{,}000$	Yes (margin $\sim 3.5\times$)
15	$\sim 2{,}800$	Yes (margin $\sim 2.0\times$)
16	$\sim 1{,}150$	No

Linear interpolation between $s = 15$ and $s = 16$ gives the critical grafting distance:

$$\boxed{s_{\max} \approx 15.9 \text{ nm}} \quad (L_0 = 4 \text{ nm},\ \sigma = 17 \text{ mN/m}). \tag{6.19}$$

At $s = s_{\max}$, the equilibrium wall-film thickness is $h_{\min} \approx 1.6$ nm, sitting at the peak of the disjoining-pressure profile. Denser coverage ($s < s_{\max}$) gives a larger $h_{\min}$ and a more robust film. R1 is satisfied for any practical sub-CMC concentration in HyDRA's operating range with a steric margin of at least $2\times$.

6.5.3 Slug–Slug Coalescence (R2)

R2 demands that two slugs parked against opposite faces of a closed valve coalesce within a bounded time after the valve opens. The argument has two parts: a static-barrier analysis showing that the equilibrium between capillary driving and steric repulsion sits inside the regime where the surfactant interface is well below stabilization saturation, and three independent experimental anchors from the droplet-microfluidics literature that place rupture in the millisecond-to-second range under structurally analogous conditions.

Valve retraction and capillary approach. Consider two slugs parked against opposite faces of a closed valve V_2 (Chapter 7, Section 7.5). When V_2 opens, the deflected membrane retracts into the control layer and flushes against the ceiling of the connector cell, freeing a channel volume of order $WH\ell_v = \gamma \approx 0.6$ nL. The four corner gutters in that cell hold only $\sim 4(1 - \pi/4)\xi_b^2 \cdot \ell_v \approx 0.025$ nL — an order of magnitude less than the freed volume. The deficit drives the two slug caps inward, each pressing into the newly available volume under cap Laplace pressure $\Delta P_{\text{cap}} \approx 1{,}473$ Pa, with a combined approach pressure of $2\Delta P_{\text{cap}} \approx 2{,}947$ Pa.

The intervening oil drains laterally through the corner gutters of the connector cell. While the valve is extended (closed), the membrane occludes the two top corner gutters of that cell completely, leaving only the two bottom gutters available for any oil flow [14]. When the membrane retracts on opening, all four corner gutters of the cell are restored, and the inter-cap film drains through this restored four-path lateral network as the caps approach. Critically, the connector-cell gutters are *unconfined* — they open directly into the bus channel at J_U and into the register at J_D — so drainage is not bottlenecked by long bypass paths.

Static disjoining-pressure barrier. The slug–slug film is a double-brush configuration: each interface carries a Fluosurf-C brush of thickness L_0, so the steric profile is set by Equation 6.9 with $L_0 \to 2L_0 = 8$ nm. The combined disjoining-pressure profile $\Pi_{\text{net}}(h) = \Pi_{\text{steric}}(h) + \Pi_{\text{vdW}}(h)$ at $s = s_{\max} = 15.9$ nm is given in Table 6.5.

Table 6.5: Net slug–slug disjoining-pressure profile at $s = s_{\max} = 15.9$ nm, $2L_0 = 8$ nm. Negative Π_{net} means net attractive (the film thins spontaneously); positive means the film is mechanically supported.

h (nm)	Π_{net} (Pa)	Note
8.0	−293	Brush tips touching; vdW attraction dominates
7.0	+17	Brush mildly compressed
5.0	+1,019	Approaching equilibrium
3.2	+2,923	$\approx h_{\text{eq}}$ at $\Pi_{\text{net}} = 2\Delta P_{\text{cap}}$
2.21	**+4,133**	**Barrier peak**
1.5	−736	vdW dominated; film about to rupture
1.0	−40,828	Rupture

The film equilibrates near $h_{\text{eq}} \approx 3.2$ nm where the steric repulsion balances the capillary driving pressure. The peak of Π_{net} at $h = 2.21$ nm sits $\sim 1{,}210$ Pa above the equilibrium pressure, which is the static barrier the film must traverse to rupture under purely thermodynamic driving. Whether that barrier traversal happens in milliseconds or in seconds is set by the kinetics of brush compression, surfactant

adsorption, and stochastic film rupture under the sustained $2\Delta P_{\text{cap}}$ load. Static disjoining-pressure analysis alone does not pin the timescale; the experimental literature does.

Coverage as the operative variable. Translating the $s_{\max} = 15.9$ nm grafting distance to an interfacial coverage $\Gamma/\Gamma_{\text{sat}}$ via PFPE-PEG adsorption isotherms on HFE-7500 interfaces [6] places the R1-derived lower edge of the operating window at $\Gamma/\Gamma_{\text{sat}} \approx 4\%$. The upper edge is a designer choice: Mazutis [28] and Riechers [34] place coalescence suppression above $\Gamma/\Gamma_{\text{sat}} \approx 90\%$, and we adopt 14% as the operating ceiling — well below the suppression threshold (margin $> 6\times$) while keeping the slug–slug barrier comfortably traversable under the capillary load. The architectural operating window is therefore

$$\Gamma/\Gamma_{\text{sat}} \in [4\%, 14\%]. \tag{6.20}$$

The numeric values depend on a per-chip calibration of Γ_{sat} at the chosen geometry and temperature, which we treat as a feasibility gate rather than a derived constant. The architecturally important fact is that the band is non-empty and finite, and that within it, the published droplet-microfluidics literature places coalescence on the millisecond-to-second timescale.

Three experimental anchors. Three independent results, taken together, place the rupture timescale in HyDRA's regime well below any architectural deadline:

- Mazutis & Griffiths [28] report that droplets at $\Gamma/\Gamma_{\text{sat}} > 98\%$ paired with droplets at $\Gamma/\Gamma_{\text{sat}} < 16\%$ fuse spontaneously upon contact with no minimum contact time required, and that suppression of passive coalescence sets in only above $\Gamma/\Gamma_{\text{sat}} \approx 90\%$. HyDRA pairs both slugs at or below 14% — outside Mazutis & Griffiths' explicitly tested combinations but strictly more favorable, since symmetric low coverage exposes bare water–oil interface on both caps, accelerating drainage and contact relative to the asymmetric case.

- Riechers *et al.* [34] measured surfactant adsorption times $\tau_{\text{ads}} \in \{3.6, 24, 120\}$ ms across a representative range of PFPE-PEG concentrations and confirmed that coalescence is suppressed only above $\sim 90\%$ surface coverage. At HyDRA's 4–14% coverage, the adsorption-replenishment timescale — the rate at which the bulk reservoir restores surfactant to a thinning interface — sits in the millisecond range, far shorter than any plausible drainage time.

- Dangla *et al.* [10] demonstrate passive coalescence on the seconds timescale at surface-energy anchors with side-bypass drainage geometry. Those longer timescales are drainage-limited: the bypass channels constrain the lateral flow path. HyDRA's connector cell exposes four unconfined corner gutters opening directly onto the bus and the register, so its drainage is bounded above by the Dangla geometry.

Together, the three results bracket HyDRA's regime: coverage well below the suppression threshold (Mazutis & Griffiths), surfactant kinetics in milliseconds (Riechers *et al.*), and a more favorable drainage geometry than the seconds-timescale precedent (Dangla *et al.*). The expected rupture timescale is therefore in the millisecond-to-second range.

Architectural budget. We schedule $t_c \sim 10$ s as the WAIT duration following any MERGE compound (Chapter 9). This is conservative by at least an order of magnitude relative to the experimental anchors; it is not the predicted timescale but a margin against the worst case where the deployed surfactant batch sits at the upper edge of the predicted range. MERGE is already the slowest single-tile compound in the ISA, and a 10 s budget does not break any target assay timeline. If on-chip characterization confirms sub-second rupture as the bracket suggests, the budget can be tightened; the architecture is robust to either outcome.

Wall-film safety during coalescence. The wall film at the parked slug must remain stable while the inter-cap film ruptures. Wall-film drainage at the narrow face has timescale ~ 58 s by Reynolds thin-film theory (Section 6.6) — more than an order of magnitude longer than the entire scheduled coalescence event. R1 and R2 do not compete on a shared margin.

6.6 Film Drainage During Parking

Sections 6.5.2 and 6.5.3 treat the wall film and the slug–slug film at their disjoining-pressure equilibrium thicknesses. Reaching equilibrium takes time. When a slug first parks — its cap pressed against a closed valve membrane — the wall film is still at the dynamic Bretherton thickness $h_\infty \approx 163$ nm (Chapter 2). The corner-suction load (Equation 6.6) drives the film to drain laterally toward the corner gutters until it reaches the equilibrium thickness $h_{\min} \approx 1.6$ nm of Section 6.5.2.

Reynolds thin-film theory [33, 35] gives the drainage timescale for a film of thickness $h_{\min}$ across a face of effective half-width R_{drain} under driving pressure ΔP_{corner}:

$$t_{\text{drain}} \approx \frac{3\mu R_{\text{drain}}^2}{4\Delta P_{\text{corner}} h_{\min}^2}. \tag{6.21}$$

For HyDRA's rectangular cross-section the slug has two pairs of faces of different widths — two narrow H-edge faces and two wide W-edge faces — with correspondingly different drainage paths. Table 6.6 reports the timescales.

Table 6.6: Wall-film drainage timescales by face. The narrow-face drainage is the relevant timescale because both faces must thin for the wall film to be unstable; the wide face is included for completeness.

Face	R_{drain}	Drainage path to nearest gutter	t_{drain}
Narrow (H-edge)	$H/2 = 15$ μm	Half-channel-height, perpendicular to long axis	~ 58 s
Wide (W-edge)	$W/2 = 50$ μm	Half-channel-width, perpendicular to long axis	~ 642 s

The wall-film stability analysis of Section 6.5.2 is therefore relevant on two timescales. For parking durations under $\sim$ 1 minute, the film is still well above $h_{\min}$ and disjoining pressure is essentially irrelevant; the slug is held in place by valve contact alone. For longer parking durations, the wall film approaches $h_{\min}$ and the steric stability margin in Equation 6.19 becomes load-bearing. R1 is engineered for the worst case — parking durations of arbitrary length — with the full $\geq 2\times$ steric margin at $s = s_{\max}$.

6.7 Joint Window and Feasibility

The three requirements R1, R2, R3 jointly define an operating window. Table 6.7 summarizes whether each requirement is met at HyDRA's chosen operating point.

Table 6.7: Feasibility summary at HyDRA's chosen operating point.

	Condition	Status
R1	$s < s_{\max} = 15.9$ nm ($\Gamma/\Gamma_{\text{sat}} \gtrsim 4\%$)	Satisfied with $\geq 2\times$ steric margin (Section 6.5.2)
R2	$\Gamma/\Gamma_{\text{sat}} \lesssim 14\%$, unconfined corner-gutter drainage	Bracketed in ms-to-second range by [28, 34, 10]
R3	$Ca < 10^{-3}$	$Ca = 7.3 \times 10^{-4}$ at v_{nom}, 7.3×10^{-5} at v_{prec}

R1 and R2 are simultaneously satisfiable inside the coverage band $\Gamma/\Gamma_{\text{sat}} \in [4\%, 14\%]$: R1 sets the lower edge (denser coverage strengthens the wall-film steric barrier), R2 sets the upper edge (sparser coverage keeps the slug–slug interface below the suppression threshold). The static effective barrier of ~ 1.2 kPa for the slug–slug film sits well below the capillary driving pressure $2\Delta P_{\text{cap}} \approx 2{,}947$ Pa, and the experimental anchors of Section 6.5.3 place coalescence in the millisecond-to-second range under HyDRA's drainage geometry. Wall-film drainage during the merge event is bounded below by ~ 58 s (Section 6.6) — more than an order of magnitude longer than any scheduled coalescence, so R1 and R2 do not compete on a shared margin. R3 is satisfied with order-of-magnitude margin at every operating velocity.

Surfactant concentration calibration. The architecturally targeted band corresponds to a sub-CMC Fluosurf-C concentration below the Gilet baseline (0.5% w/w [14]); the typical operating range is 0.02–0.10% w/w. Empirical commissioning of a fabricated chip therefore requires four steps:

1. Map the adsorption isotherm $\Gamma([c])$ via pendant-drop tensiometry and the Gibbs adsorption equation, translating the architectural coverage band to a bulk surfactant concentration.
2. Titrate the surfactant concentration downward from 0.5% w/w; identify the concentration at which spontaneous coalescence at a closed-then-opened valve first succeeds on the deployed device.
3. Confirm wall-film integrity at the operating concentration during prolonged parking (> 10 minutes), ruling out long-timescale R1 violation.
4. Confirm coalescence is insensitive to the aqueous-phase ionic strength of the target assay, since the disjoining-pressure model assumes the EDL term is absent — valid for fluorocarbon oils but worth checking with a representative reagent.

6.8 Summary

Table 6.8 collects the regime parameters that downstream chapters reference. All values are computed at $L_0 = 4$ nm, $\sigma = 17$ mN/m, $s = s_{\max}$, and are conditional on per-device measurement of L_0 and the adsorption isotherm $\Gamma([c])$.

Table 6.8: Regime summary — representative values at the chosen operating point.

Parameter	Value
Fluid system	HFE-7500 / Fluosurf-C (sub-CMC) / Aquapel-PDMS
Corner-meniscus radius ξ_b	≈ 12 μm
Capillary number Ca (nominal)	7.3×10^{-4}
Reynolds number Re	0.39
Bretherton film h_∞	≈ 163 nm
Slug drift bound $\lvert\alpha\rvert$	≤ 0.03
Hamaker A_{123} / A_{323}	4.8×10^{-22} J / 2.83×10^{-21} J
Asymmetry A_{323}/A_{123}	≈ 5.9
Cap pressure ΔP_{cap}	$\approx 1{,}473$ Pa
Corner-suction ΔP_{corner}	$\approx 1{,}417$ Pa
Coalescence drive $2\Delta P_{\text{cap}}$	$\approx 2{,}947$ Pa
Effective slug–slug barrier	~ 1.2 kPa
Critical grafting distance $s_{\max}$	≈ 15.9 nm
Coverage band $\Gamma/\Gamma_{\text{sat}}$	4–14%
Wall-film equilibrium $h_{\min}$	≈ 1.6 nm
Slug–slug equilibrium h_{eq}	≈ 3.2 nm
Coalescence timescale (literature anchors)	ms–s [28, 34, 10]
Architectural budget t_c	~ 10 s (conservative)
Wall-film drainage (narrow / wide face)	~ 58 s / ~ 642 s

The fluid system satisfies all three requirements simultaneously. R1 is over-engineered at any practical sub-CMC concentration: the steric barrier exceeds the corner-suction load by a factor of two or more across the entire 4–14% coverage band. R2 is anchored in three independent experimental results that bracket HyDRA's regime — coverage well below the suppression threshold (Mazutis & Griffiths), surfactant kinetics in milliseconds (Riechers *et al.*), and a more favorable drainage geometry than the seconds-timescale precedent (Dangla *et al.*) — placing rupture in the millisecond-to-second range, with the architectural budget $t_c \sim 10$ s carrying an order-of-magnitude safety margin. R3 is trivially satisfied at every operating velocity. The two operative empirical unknowns — the surfactant brush thickness L_0 and the per-device adsorption isotherm $\Gamma([c])$ — are both closed by pendant-drop tensiometry and on-chip titration during commissioning. With these calibrations in place, the abstract cell-grid model of Chapter 5 stands on a physically realizable substrate.

Chapter 7

Physical Architecture

This chapter specifies the physical layout that hosts the discrete computational model of Chapter 5 and operates inside the regime of Chapter 6. The architecture is a closed loop with a single peristaltic pump, ten serpentine register tiles, an on-loop mixing contour, a four-port I/O zone, and a pneumatic demultiplexer that addresses every flow-layer valve. With the pump quantum, slug regime, and fluid system fixed by the previous two chapters, this chapter pins the geometry, valve assignments, tile-mode dictionary, and loop dimensions on which the operations of Chapter 9 and the ISA of Chapter 10 are built. Uniform channel cross-section and a single cell length ℓ_v throughout make one pump stroke advance every fluid element on the active path by exactly one cell.

7.1 Overview

HyDRA (Figure 7.1) is a closed-loop, single-bus, two-phase microfluidic chip, fabricated as a three-layer PDMS multilayer device by soft lithography [42]. The flow layer carries aqueous reagent slugs separated by HFE-7500 carrier oil; a gain-valve layer realizes a Grover pressure-amplification array [17] that drives the working valves; and a demultiplexer layer addresses individual gain valves through a binary address tree [41]. Ten register tiles $T_0, \ldots, T_9$ are distributed around a circular bus that passes, in clockwise order, through the peristaltic pump, the ten tiles, the mixing contour, and the I/O zone before returning to the pump inlet. Every tile exposes one off-chip module port through which an external processing unit (e.g. a thermocycler, an absorbance reader, or a fluorescence detector) interacts with that tile's register contents.

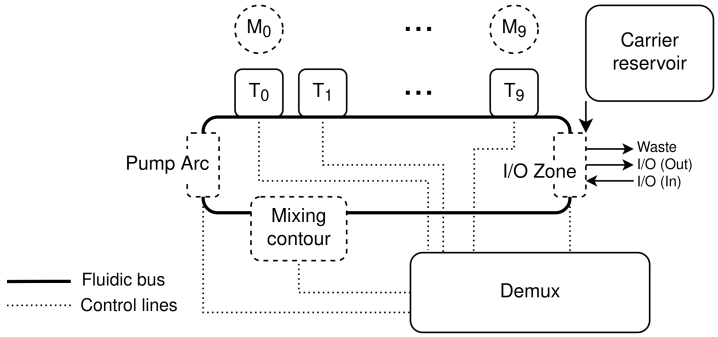


Figure 7.1: HyDRA system overview. Ten register tiles $T_0, \ldots, T_9$ sit on a closed bus served by a single peristaltic pump. Each tile T_i exposes an off-chip module port M_i for external processing hardware. The I/O zone groups reagent inlet, waste outlet, and sample collection; the carrier reservoir keeps the loop primed. The mixing contour is a bus-side branch. All valves are addressed through a shared 7-bit demultiplexer (dotted control lines).

The architectural parameters in Table 7.1 are referenced throughout this chapter. They fix the spatial discretization, register sizing, and loop dimensions; downstream chapters depend on them but never

re-derive them.

Table 7.1: Architectural parameters.

Symbol	Value	Description
N	10	Register count
V_{reg}	416 cells $\approx$ 250 nL	Register capacity
L_{bus}	619 cells	All-Bypass bus length
C_{loop}	123.8 mm	Loop circumference
ℓ_v	200 μm	Cell length
$W \times H$	100×30 μm	Channel cross-section
γ	≈ 0.6 nL	Pump quantum
v_{nom}	10 mm/s (50 Hz)	Nominal velocity
v_{prec}	1 mm/s (5 Hz)	PARK precision velocity
σ	17 mN/m at CMC	Oil–water IFT
n_{XC}	≤ 2	Compliance bound

7.2 Standards Compliance

The chip conforms to ISO 22916:2022, the international standard for microfluidic device interoperability [21]. Adopting the standard yields three engineering benefits without restricting the architecture: off-chip tubing, clamps, and manifolds are commercial off-the-shelf items; external modules from established modular microfluidic platforms [39] interoperate without bespoke plumbing; and the chip is physically compatible with standard microscope stages and imaging hardware.

The chip diameter $d_{\text{chip}} \approx 44$ mm (Section 7.8) fits inside the ISO credit-card format (84×54 mm), with margin for port routing, clamping zones, and pneumatic control-layer headers. Every off-chip interface — the four I/O ports and the ten module-side ports — follows the ISO 22916 grid: 3 mm port pitch on the chip's long edge, 0.75 mm internal-diameter press-fit tubing [21], and a 6×6 mm exclusion zone per port. The application class is PT 2/100: maximum fluid pressure 2×10^5 Pa and operating temperature 4–100 °C, both of which accommodate HyDRA's nominal 20–50 kPa flow regime and the temperature range from enzymatic-assay working temperatures to PCR thermal cycling.

7.3 Channel Geometry and Cell Discretization

Every channel on the chip — bus arc, register serpentine, bypass connector, mixing contour, I/O stub, module port — has the same rectangular cross-section $W \times H = 100 \times 30$ μm, the same as the cell-grid geometry of Chapter 5. Uniformity is load-bearing: the hydraulic-transmission invariant requires that pump displacement convert to fluid advance through a single conserved cross-sectional area; any local widening or narrowing would create capacitance that decouples the stroke count from the cell index. The aspect ratio $W/H \approx 3.3$ is deliberate, ensuring four Bretherton corner gutters per cross-section (Chapter 6, Section 6.3.2) so that a closed valve occludes the slug body while letting carrier oil bypass laterally. PDMS walls are treated with Aquapel to render the surface fluorophilic (Chapter 6, Section 6.3.1).

A *cell* is the channel segment of length $\ell_v = 200$ μm and cross-section $W \times H$; a single closed Quake valve occupies exactly one cell, displacing the volume quantum $\gamma = WH\ell_v = 0.6$ nL per cycle of the peristaltic pump (Chapter 5). All channel lengths, slug volumes, and slug positions on the chip are integer multiples of ℓ_v and γ respectively. This is what allows the abstract state of Chapter 8 to address physical positions exactly: a slug at abstract cell index x occupies the physical channel segment $[x \cdot \ell_v, (x+1) \cdot \ell_v]$ on the active flow path.

7.4 Register

A register stores one reagent slug. The architecture provides ten registers, each instantiated as a folded uniform-cross-section serpentine that fits in a compact bounding box. The capacity is set by the target

assay class; the geometry is set by the slug-integrity threshold for two-phase flow in curved channels.

7.4.1 Volume Target

A register must hold at least one complete reagent slug for the target assay class. Table 7.2 surveys representative single-reagent volumes drawn from the microfluidic biochemistry literature; the architecture's target capacity is set to cover all five assays without scaling.

Table 7.2: Representative single-reagent volumes for the target assay class.

Assay	Volume (per reagent)	Reference
PCR (droplet)	6.5–280 nL	Ottesen *et al.* [32]; Lagally *et al.* [25]
Immunoassay	$\sim$ 100 nL	Kartalov *et al.* [24]
Serial dilution	$\sim$ 400 nL total	Grover *et al.* [16]
Protein crystallization	10–100 nL	Hansen *et al.* [20]
Glucose assay	50–200 nL	Srinivasan *et al.* [36]

A nominal target of 250 nL covers all single-reagent cases in Table 7.2. Setting $V_{\text{reg}} = 416$ cells gives a nominal capacity

$$V_{\text{reg}} \cdot \gamma = 416 \times 0.6 \text{ nL} = 249.6 \text{ nL}, \tag{7.1}$$

within 0.2% of the target. The physical serpentine occupies exactly V_{reg} cells; safe-closure clearance between a parked slug and the mouth valves V_1, V_2 (Section 7.5) is enforced operationally by the runtime (Chapter 9), not by additional lithographic margin.

The physical volume held in a register depends on the per-chip γ_{actual} (Chapter 5, Section 5.1.2): from 170 nL at the worst-case low ($\gamma_{\text{actual}} = 0.68\gamma$) to 332 nL at the worst-case high ($\gamma_{\text{actual}} = 1.33\gamma$). The compiler's calibration scalar maps user-specified target volumes to integer cell counts, ensuring no slug ever requires more than V_{reg} cells regardless of γ_{actual}. Chips at the extreme low end of γ_{actual} may not be able to host the full 250 nL nominal target on a single register: this is a per-chip yield consideration, not a specification failure.

7.4.2 Serpentine Geometry

The register channel folds a uniform-cross-section channel of length

$$L_{\text{serp}} = V_{\text{reg}} \cdot \ell_v = 416 \times 200 \ \mu\text{m} = 83.2 \text{ mm} \tag{7.2}$$

into a compact rectangular serpentine. At the operating capillary number $Ca = 7.3 \times 10^{-4}$ (Chapter 6), surface tension dominates viscous forces and slug-cap geometry is independent of channel curvature, provided the bend radius respects a slug-integrity threshold.

Bend radius. The slug-integrity threshold for two-phase flow in rectangular microchannels is $R \geq 2W$ [46, 19]; below this threshold, the slug deforms or breaks at the bend. HyDRA uses $R = 300$ μm $= 3W$, comfortably above the threshold while keeping the U-turn footprint small.

Pass count. Eight vertical passes connected by seven U-turns. The even pass count places both serpentine endpoints on the same side of the bounding box, giving a symmetric inlet/outlet pair at the connector valves V_1 and V_2 (Section 7.5). The seven U-turns contribute $7\pi R = 6.6$ mm of curved channel.

Pass dimensions. The remaining $L_{\text{serp}} - 7\pi R = 83.2 - 6.6 = 76.6$ mm of straight channel, distributed across eight passes, gives a pass height of $76.6/8 = 9.6$ mm. With seven inter-pass U-turns of diameter $2R = 600$ μm each, plus the channel width W at the outermost pass, the serpentine width is

$$W_{\text{serp}} = 7 \times 2R + W = 4.3 \text{ mm}. \tag{7.3}$$

The inter-pass wall thickness (PDMS between adjacent straight passes) is $2R - W = 500$ μm, well above the PDMS fabrication minimum of ~ 50 μm [42].

Bounding box. Including a fabrication margin around the active geometry, the serpentine occupies a bounding box of 5×10 mm. Across all ten tiles this consumes 500 mm^2 of flow-layer area, within the annular region available (Section 7.8). Further geometric optimization is possible but not pursued; this is a minimum viable layout.

Table 7.3: Serpentine geometry summary.

Parameter	Value
Bend radius R	300 μm ($3W$, above $2W$ slug-integrity threshold)
Passes	8 vertical (7 U-turns; even count for symmetric endpoints)
Turn length	$7\pi R = 6.6$ mm
Pass height	$(L_{\text{serp}} - 7\pi R)/8 = 9.6$ mm
Serpentine width	$7 \times 2R + W = 4.3$ mm
Bounding box	5×10 mm (with fabrication margin)

7.5 Register Tile

A register tile is the fundamental operational unit of the architecture. It bundles one register serpentine with the switching valves and module-side port that connect it to the bus and to off-chip processing hardware. The tile is the address granularity of the ISA: every load, unload, merge, and process targets a tile by index. Figure 7.2 shows the tile schematic and labels the seven flow-layer valves used throughout this section.

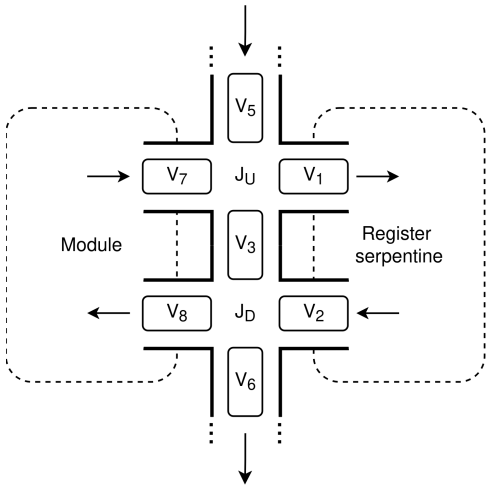


Figure 7.2: Register tile schematic. The two cross-junctions J_U (upstream) and J_D (downstream) gate the four channel arms — bus, bypass, register, and module port — with valves V_5/V_6 on the bus, V_3 on the bypass, V_1/V_2 on the register inlet/outlet, and V_7/V_8 on the module port. The register serpentine connects J_U to J_D through V_1 and V_2. Off-chip module ports are denoted with circles; bus arrows indicate clockwise flow.

7.5.1 Junction Structure

Each tile has two four-way cross-junctions on the bus, denoted J_U (upstream, clockwise) and J_D (downstream, counter-clockwise). Four channel arms meet at each junction: the bus, the bypass channel, the register connector, and the module-side port. Table 7.4 enumerates the arms and their gating valves.

Table 7.4: Channel arms and gating valves at the upstream and downstream junctions of a register tile.

Arm	At J_U	At J_D
Bus	Upstream segment, gated by V_5	Downstream segment, gated by V_6
Bypass	To J_D, gated by V_3	From J_U, same V_3, same cell
Register	Inlet connector, gated by V_1	Outlet connector, gated by V_2
Module	Off-chip port, gated by V_7	Off-chip port, gated by V_8

Each junction is physically a $W \times W$ intersection (100×100 μm). To preserve the uniform ℓ_v-cell grid across the bus, each junction absorbs 100 μm of adjacent bus channel: J_U takes its 100 μm from the upstream bus arm, J_D from the downstream bus arm. Intersection-plus-bus-segment together forms a single ℓ_v-long cell. Without this absorption, junctions would create irregular short cells and break the stroke-to-cell correspondence. The absorption is fixed by lithography, so the tile contributes exactly one bus cell at J_U and one at J_D regardless of mode.

The bypass channel between J_U and J_D is itself a single ℓ_v-long cell, spanned entirely by the valve V_3: the connector and its valve are the same physical cell, so closing V_3 occludes the entire bypass with no trapped fluid and opening V_3 exposes a one-cell direct connection across the tile.

7.5.2 Register Path

The register connects J_U to J_D via two connector cells and the serpentine. Each connector is a single ℓ_v-long cell spanned entirely by a gate valve — V_1 at the inlet (from J_U) and V_2 at the outlet (to J_D). The connector and its valve are the same physical cell; there is no channel between the junction and the valve. When a gate valve closes, it occludes the entire connector with no trapped fluid (Chapter 5). The full tile composition along the active path, with the register engaged, is given in Table 7.5.

Table 7.5: Full tile cell count along the active path with the register engaged (Exchange mode).

Element	Cells	Role
J_U	1	Upstream junction cell (fixed; always on path)
V_1 (inlet gate)	1	Inlet connector; valve spans the full cell
Serpentine	416	Register storage; V_{reg} cells ≈ 250 nL
V_2 (outlet gate)	1	Outlet connector; slug anchor point
J_D	1	Downstream junction cell (fixed; always on path)

The junction cells J_U, J_D are on the active path regardless of tile mode. The switchable segment between them is either V_3 alone in Bypass mode (1 cell) or V_1 + serpentine + V_2 in Exchange mode ($V_{\text{reg}} + 2$ cells); the difference is the per-tile mode-switch shift formalized in Section 7.7.2.

7.5.3 Module Ports

Each tile exposes one off-chip module port through two channels gated by V_7 (from J_U) and V_8 (from J_D). These radiate outward from the bus to a punched through-hole port at the chip edge approximately 2 mm radially out from the junction. Press-fit tubing of standard 0.75 mm internal diameter (23-gauge stub) [21] couples the on-chip port to whatever external processing element the application provides. A single tile binds to a single module — a thermocycler, an absorbance reader, a fluorescence detector, an HPLC sampling loop — and the architecture treats the module's behavior as an opaque contract: input volume in, output volume out, after a fixed wait. Multiple inputs to one module are realized by serially loading and merging slugs in the register before transfer, not by adding physical ports (Chapter 10, Section 10.5). The bus–register coupling switches in and out of the module path via the Module standard mode (Section 7.5.5).

7.5.4 Tile Valves

Seven Quake push-down membrane valves [42] per tile collectively define the tile's mode and gate every fluid path through it. Each valve occupies exactly one cell. Table 7.6 enumerates them.

Table 7.6: Tile valve assignments. All seven valves are Quake push-down membranes spanning one cell each.

Valve	Location	Function
V_1	$J_U \rightarrow$ register	Inlet gate; spans the inlet connector cell
V_2	register $\rightarrow J_D$	Outlet gate and slug anchor; spans the outlet connector cell
V_3	$J_U \leftrightarrow J_D$	Bypass gate; spans the single bypass cell
V_5	bus $\rightarrow J_U$	Bus upstream isolation
V_6	$J_D \rightarrow$ bus	Bus downstream isolation
V_7	$J_U \rightarrow$ module	Module port gate (upstream side)
V_8	$J_D \rightarrow$ module	Module port gate (downstream side)

7.5.5 Standard Modes

The tile has three *standard* operating modes — BP (Bypass), XC (Exchange), and MD (Module) — between which it can sit indefinitely with the pump idle or running. Standard modes are the rest states of the architecture: every compound ISA operation begins and ends in a standard mode (Chapter 9). Table 7.7 gives the valve configuration of each.

Table 7.7: Standard tile modes. Valve states encoded as 1 (open) and 0 (closed).

Mode	V_1, V_2	V_3	V_5, V_6	V_7, V_8	Active path through tile
BP	0	1	1	0	Bus $\rightarrow J_U \rightarrow$ bypass $\rightarrow J_D \rightarrow$ Bus
XC	1	0	1	0	Bus $\rightarrow J_U \rightarrow V_1 \rightarrow$ register $\rightarrow V_2 \rightarrow J_D \rightarrow$ Bus
MD	1	0	0	1	Module $\rightarrow J_U \rightarrow$ register $\rightarrow J_D \rightarrow$ Module

Bypass (BP) is the default mode and the unpowered rest state of the chip: the lithographic preload of each working valve is chosen so that, with all pneumatic pressures vented, the unactuated tile sits in BP. The register is sealed and any slug parked inside is anchored against V_2; bus traffic flows through the one-cell bypass. The tile contributes 3 cells to the active path ($J_U + V_3 + J_D$).

Exchange (XC) couples the register serpentine into the active path: bus flow passes through V_1, the full V_{reg}-cell serpentine, and V_2 before continuing past J_D. The tile contributes $J_U + V_1 + V_{\text{reg}} + V_2 + J_D = 420$ cells to the active path. XC is the mode used to load a slug from the bus into the register, to unload a register slug back onto the bus, and to merge two slugs that share a register (Chapter 9).

Module (MD) isolates the tile from the bus and couples the register to the off-chip module. With V_5, V_6 closed, the tile's local subnetwork — J_U, the register, J_D, and the module port — is fluidically disconnected from the bus, and V_3 remains closed so no carrier oil leaks between the module circuit and the bus. While any tile is in MD the bus loop is broken at that tile, so the bus pump must be held idle: bus-side operations resume only when the tile returns to BP. Asynchronous module invocation, defined at the ISA level (Chapter 10), brings the tile back to BP between the short transfer phases that flank the long off-chip processing wait, so the bus is paused only during transfer.

7.5.6 Transient Modes

Two further valve configurations, AC (All-Closed) and RP (Register-Park), arise transiently within compound operations. They are not steady states: the scheduler enters them only with the pump idle, performs at most one CIRC under the configuration, and exits to a standard mode before the next compound operation begins (Chapter 9, Section 9.6). Each relaxes one of the standard-mode constraints (Section 7.5.7) and so requires individual-valve addressing through the demultiplexer.

Table 7.8: Transient tile configurations. Both occur only within compound operations and exit before the operation completes.

Mode	V_1	V_2	V_3	V_5, V_6	V_7, V_8	Use
AC	0	0	0	1	0	MERGE pre-park; LOAD/UNLOAD intermediate
RP	1	0	0	1	0	In-register parking (LOAD, UNLOAD)

All-Closed (AC) seals every internal flow path through the tile while leaving the bus connection live ($B = 1$, $V_3 = 0$, $R = 0$). AC arises naturally as the midpoint of a BP $\rightarrow$ XC transition under the break-before-make discipline (Section 7.5.8) — after V_3 closes but before V_1, V_2 open. The MERGE compound exploits this midpoint: with the register sealed by V_1, V_2 and the bus broken at V_3, a bus slug can be parked against the outer face of V_2 while the register slug remains undisturbed inside, setting up two-sided coalescence when the next mode transition opens V_2 (Chapter 9, Section 9.5).

Register-Park (RP) opens the inlet alone, leaving the outlet sealed ($V_1 = 1, V_2 = 0$, $V_3 = 0$, $B = 1$). A CIRC step in this configuration drives carrier oil from the bus into the register through V_1, pushing the register slug against the closed V_2 membrane. After enough strokes the slug cap contacts V_2, at which point the slug's abstract position is lithographically defined: this is the re-registration mechanism that anchors the discrete model to physical hardware. RP is used during LOAD (to seat a freshly transferred slug at V_2) and during UNLOAD (to push a register slug forward against V_2 before opening it onto the bus). The compound-op semantics and stroke counts are detailed in Chapter 9.

7.5.7 Control Simplification

The seven flow-layer valves of a tile expose $2^7 = 128$ raw configurations, of which the architecture uses only five: three standard modes and two transient modes. The reduction follows from structural relations on the valve state, not from a chosen subset of the configuration space.

Pairings. In every standard mode, three valve pairs are jointly addressed:

$$V_1 = V_2, \qquad V_5 = V_6, \qquad V_7 = V_8 \qquad \text{(standard modes)}. \tag{7.4}$$

Define the paired variables

$$R := V_1 = V_2 \quad \text{(register gate)}, \qquad B := V_5 = V_6 \quad \text{(bus gate)}, \qquad M := V_7 = V_8 \quad \text{(module gate)}. \tag{7.5}$$

The standard-mode tile state collapses from seven valves to four bits $(R, V_3, B, M) \in \{0,1\}^4$.

Complementarity. Two further constraints hold across the standard modes:

$$V_3 = \neg R, \qquad M = \neg B \qquad \text{(standard modes)}. \tag{7.6}$$

The first encodes *register/bypass exclusion*: of the two parallel J_U-to-J_D paths through the tile (register and one-cell bypass), exactly one is open at any standard-mode instant. The second encodes *bus/module exclusion*: the tile couples either to the bus or to the off-chip module, never both. Together (7.4) and (7.6) reduce the standard-mode tile state to two free bits $(R, B) \in \{0,1\}^2$. Of the four candidate configurations, the architecture excludes $(R, B) = (0, 0)$ — a route from bypass to module that bypasses the register without coupling to the bus, with no architectural use — leaving the three standard modes:

$$\text{BP} = (R, B) = (0, 1), \qquad \text{XC} = (1, 1), \qquad \text{MD} = (1, 0). \tag{7.7}$$

Transient relaxation. Each transient mode relaxes exactly one of the constraints (7.4), (7.6):

AC violates the register-side complementarity $V_3 = \neg R$ by setting $R = V_3 = 0$, closing both J_U-to-J_D paths and isolating the register from a bus slug parked against V_2 at the connector face.

RP violates the pairing $V_1 = V_2$ by setting $V_1 = 1, V_2 = 0$, opening the register inlet alone so that bus-driven flow re-registers the in-register slug against the closed outlet membrane.

Both transients require individual addressing of V_1 or V_2 via the demultiplexer (Section 7.6.4). The scheduling discipline of Chapter 9 enforces (7.4)–(7.6) per tile in steady state and admits each transient only inside a compound operation, with the pump idle and a single CIRC step under the relaxed configuration.

Cross-tile mode vector. For a chip with N register tiles, the steady-state configuration is the vector

$$C = (c_0, c_1, \ldots, c_{N-1}) \in \{\mathsf{BP}, \mathsf{XC}, \mathsf{MD}\}^N, \tag{7.8}$$

where c_i is the standard mode of tile T_i. C is the only tile-level state visible to the formal abstraction of Chapter 8, which inspects the chip only between completed ISA operations. Transient configurations (AC, RP) are interior to compound steps and are not part of C. The mixer state $s_{\text{mix}} \in \{0, 1\}$ and I/O zone state $s_{\text{IO}} \in \{\mathsf{ID}, \mathsf{IN}, \mathsf{FL}, \mathsf{EJ}\}$, defined in Sections 7.6.2 and 7.6.3, complete the steady-state valve configuration of the chip.

7.5.8 Mode Transitions

Mode transitions follow a *break-before-make* discipline: the departing path closes before the arriving path opens. This ensures that no instant of the transition exposes two simultaneously open paths through the tile, which would create an undefined branching of the active path. Table 7.9 lists the four standard transitions and their valve-change sequences.

Table 7.9: Standard mode transitions and their valve-change sequences.

Transition	Step 1	Step 2
$\mathsf{BP} \to \mathsf{XC}$	Close V_3 (enter AC)	Open V_1, V_2
$\mathsf{XC} \to \mathsf{BP}$	Close V_1, V_2 (enter AC)	Open V_3
$\mathsf{BP} \to \mathsf{MD}$	Close V_3, V_5, V_6	Open V_1, V_2, V_7, V_8
$\mathsf{MD} \to \mathsf{BP}$	Close V_1, V_2, V_7, V_8	Open V_3, V_5, V_6

A $\mathsf{BP} \leftrightarrow \mathsf{XC}$ transition toggles three valves (V_1, V_2, V_3), giving a transition latency of $3\,t_v = 360$ ms at the demultiplexer's $t_v = 120$ ms per valve (Chapter 9, Section 9.6). MD transitions toggle four additional valves (V_5, V_6, V_7, V_8); the longer latency is amortized over the seconds-to-minutes time scale of the module operation that follows.

Volume redistribution. A $\mathsf{BP} \leftrightarrow \mathsf{XC}$ transition changes the active-path length by $\Delta = 417$ cells (≈ 250 nL). Because the pump is idle throughout, this change must be absorbed by the closed loop's wall compliance and the carrier oil's bulk compressibility (Chapter 5, Section 5.2.3); the analysis there bounds $\delta V_{\text{fluid}}/\gamma < 10\%$ for at most $n_{\text{XC}} \leq 2$ tiles in Exchange simultaneously, which is the architectural compliance bound.

Valve membrane displacement. When a Quake valve closes, the deflecting membrane displaces approximately $\gamma/2$ from the cell volume into the corner gutters of the adjacent channel; the reverse occurs on opening. These displacements are sub-cell and bounded; they do not affect slug positions at the cell-grid level but they do contribute to the pump-quantum variance analyzed in Chapter 5.

7.5.9 Active-Path Segment Lengths

The tile's contribution to the active path depends only on whether it is in Bypass or in Exchange (MD removes the tile from the bus entirely). Table 7.10 fixes the segment lengths.

Table 7.10: Tile contribution to the active flow path by mode.

Mode	Full tile path	Switchable segment	Switchable cells
BP	$J_U + V_3 + J_D$	V_3	1
XC	$J_U + V_1 + \text{serpentine} + V_2 + J_D$	$V_1 + \text{serpentine} + V_2$	418

7.6 Bus Components

The bus is the closed loop that connects the ten register tiles through the peristaltic pump and the on-loop service components: a mixing contour and a four-port I/O zone. We adopt a fixed clockwise ordering — pump $\to T_0 \to T_1 \to \cdots \to T_9 \to$ mixer $\to$ I/O zone $\to$ pump — so that "forward" (the + direction) is unambiguous and the path-composition equations in Section 7.7 have a canonical form. The non-register components together occupy 11 mm of bus arc; the rest is tile arcs and inter-tile gaps.

7.6.1 Peristaltic Pump

A single Goulpeau three-valve peristaltic pump [15] on a dedicated 5 mm bus arc drives the entire loop, displacing one volume quantum γ per cycle (Chapter 5). At the nominal cycle rate of 50 Hz this gives $v_{\text{nom}} = 10$ mm/s on the active path; at the precision rate of 5 Hz, $v_{\text{prec}} = 1$ mm/s used for the final approach phase of PARK (Chapter 9). The pump's three working valves are addressed through the same Grover demultiplexer (Section 7.6.4) as the rest of the chip; no dedicated controller is required.

Rotary peristaltic pumps [9] are the principal alternative. The three-valve linear pump is selected because each cycle is a single discrete stroke of volume γ: the rotary alternative delivers continuous flow and would require time-resolved transit accounting rather than stroke counting, breaking the cell-grid abstraction the rest of the chip rests on.

7.6.2 Mixing Contour

The mixing contour is a bus-side branch that, when active, routes the active path through a region of enhanced mixing geometry; when isolated, the bus continues through a one-cell bypass and the contour is sealed. The contour occupies a 2 mm bus arc and is gated by three valves arranged as a three-state T-junction: a bypass valve on the main bus, an inlet valve to the contour, and an outlet valve from the contour. The Boolean state $s_{\text{mix}} \in \{0, 1\}$ commands these three valves jointly under the SETMX primitive (Chapter 9). The contour exists because the merge operation (Chapter 10) leaves two coalesced slugs as a stratified two-layer volume; lateral diffusion alone takes ~ 10 min to homogenize the slug across the channel cross-section, longer than most assays can absorb.

The internal geometry of the mixing region is intentionally left open at the architectural level. Several PDMS-compatible options exist, summarized in Table 7.11, and the choice is application-dependent; the architecture only requires that the contour have inlet and outlet ports compatible with the bus cross-section and be controllable through the three-valve T-junction.

Of these, the Stroock herringbone variant [37] is the natural baseline: it requires no additional active valves, achieves complete cross-channel mixing in ~ 30 mm of channel at $Re < 1$, and is fabricated by a single additional photolithography step on the flow-layer mold. The rotary-loop and segmented-loop variants offer programmable mixing intensity at the cost of additional valves.

7.6.3 I/O Zone

The I/O zone provides all external fluid exchange: introducing fresh reagent slugs onto the bus, expelling completed sample slugs to off-chip collection, and replenishing carrier oil to the loop. Four gated stub

Table 7.11: Implementation options for the mixing contour. All are PDMS/Quake-compatible and integrate with the three-valve T-junction interface.

Option	Mixing mechanism	Reference
Rotary loop	Passive recirculation drives advective dispersion within the slug	Chou *et al.* [9]
Herringbone-grooved channel	Staggered grooves induce chaotic advection at low Re	Stroock *et al.* [37]
Segmented loop with sub-loop valves	Programmable recirculation paths driven by the same demultiplexer	Amin *et al.* [39]

channels branch off a 4 mm contiguous segment of the bus, each terminating in a through-hole port that connects to an off-chip sealed vial via press-fit tubing (Section 7.2). Three bus-segment valves between consecutive stubs allow the bus to be selectively broken, forcing the pump's positive displacement through a chosen pair of stubs. The bus ordering of the I/O components, in the + pump direction, is fixed to give a canonical sequence of intervening segment valves:

$$\text{bus} \to \underbrace{\text{waste}}_{V_{\text{wo}}} \xrightarrow{V_A} \underbrace{\text{reagent in}}_{V_{\text{in}}} \xrightarrow{V_B} \underbrace{\text{carrier in}}_{V_{\text{ci}}} \xrightarrow{V_C} \underbrace{\text{reagent out}}_{V_{\text{ro}}} \to \text{bus}. \tag{7.9}$$

The seven I/O zone valves — four stub valves $V_{\text{wo}}, V_{\text{in}}, V_{\text{ci}}, V_{\text{ro}}$ and three bus-segment valves V_A, V_B, V_C — are addressed individually through the demultiplexer but commanded jointly by the SETIO primitive, which exposes only the four operational modes given in Table 7.12.

Table 7.12: I/O zone configurations. Idle is the rest state with all stub valves closed and all bus-segment valves open.

Mode	Bus valve closed	Open stubs	Pump dir.	Effect per stroke
Idle (ID)	—	—	—	Bus flows straight through; no exchange
Inject (IN)	V_A	$V_{\text{wo}}, V_{\text{in}}$	+	γ reagent enters; γ carrier exits to waste
Flush (FL)	V_B	$V_{\text{wo}}, V_{\text{ci}}$	+	γ fresh carrier enters; γ old fluid exits to waste
Eject (EJ)	V_C	$V_{\text{ci}}, V_{\text{ro}}$	−	γ carrier enters; γ slug material exits to collection

Idle. All stub valves closed, V_A, V_B, V_C open — the bus passes through the I/O zone unimpeded. This is the rest state for any program point that is not currently exchanging fluid with the outside.

Inject. Closing V_A breaks the bus between waste and reagent stubs. Opening V_{wo} and V_{in} creates an L-shaped active path: from the upstream side of V_A, out through V_{wo} to the waste vial; from the reagent vial through V_{in} to the downstream side of V_A. Each forward pump stroke pushes γ of carrier oil out of the bus to waste through V_{wo} and pulls γ of reagent into the bus from the reagent vial through V_{in}. After n strokes, an n-cell reagent slug has been introduced onto the bus immediately downstream of V_A.

Flush. Closing V_B breaks the bus between the reagent inlet and the carrier inlet. The active path now drains the bus to waste while admitting fresh carrier from the carrier vial. Flush is used to clear residual reagent from the bus and to top up carrier loss from inject and eject cycles.

Eject. Closing V_C breaks the bus between the carrier inlet and the reagent outlet. With the pump running in reverse (the only place in the architecture where reverse pumping is used), γ of carrier flows in through V_{ci} and γ of slug material exits through V_{ro} to the collection vial. The slug to be ejected must already be parked at the I/O outlet position before EJ engages; this is the responsibility of the compiler, which inserts a PARK to x_{IO} before any eject (Chapter 9).

The four I/O modes encode in 2 bits and are commanded jointly by SETIO. The pump-quantum precision $|\varepsilon| \leq 0.01$ (Chapter 5, Section 5.1.2) gives a $\pm 1\%$ volume accuracy at slug creation, not amplified by subsequent transport.

7.6.4 Pneumatic Demultiplexer

Eighty-three individually addressable working valves (10 tiles × 7 each, plus the mixer's 3, the I/O zone's 7, and the pump's 3) are addressed through a single Grover-style binary pneumatic demultiplexer [17] in the control layer. The demultiplexer is a 128-output (2^7) tree of pressure-gain stages: 7 binary address inputs select one of the working valves, and a high-pressure broadcast line drives the selected valve. The selected output toggles in ~ 120 ms per valve — the latch time of one Grover stage — and this t_v is the architectural scheduling bottleneck (Chapter 9, Section 9.6).

Sizing. A 7-bit demux (128 outputs) is over-provisioned for 83 valves: a 5-bit demux (32 outputs) would suffice if every valve required individual addressing, with a 7-bit tree extrapolating Grover's 4-bit demonstration [17] along the same construction principle (binary pressure-gain trees). Two design pressures motivate the over-provisioning. First, the standard-mode pairings of Section 7.5.7 reduce the steady-state pneumatic signal count to ~ 24 across the chip, leaving spare bits for future architectural additions and redundant addressing. Second, the 7-bit binary address tree tiles in PDMS without exotic geometry. Off-chip, the demultiplexer requires 10 pneumatic lines: 7 address bits, one shared high-pressure broadcast line, one vacuum return, and one pilot pressure for the gain stages.

Footprint. The demultiplexer resides entirely in the control layers and does not consume flow-layer area. The ~ 80 latch cells, ~ 80 gain valves, and the address tree together occupy approximately 72 mm^2 of control-layer area. With the chip's central control-layer area being approximately 1,600 mm^2 (Section 7.8), the demultiplexer is $\lesssim 5\%$ of available area. The remaining control-layer area carries the inter-tile pneumatic routing: 7 address lines plus 3 shared lines per tile, at ~ 100 μm pitch, requiring ~ 1 mm of trace width per tile bundle — well within the ~ 12 mm inter-tile gap (Section 7.8.2).

7.7 Active Flow Path

At any instant, the valve configuration determines a single closed one-dimensional path around the loop — the *active path*. One pump stroke advances every fluid element on this path by exactly one cell; fluid in gated-off branches (the register serpentine of any tile in BP mode, the contour body of an idle mixer, the four stub channels of an idle I/O zone) does not move. The active-path concept is what makes the cell-grid model of Chapter 5 coherent: the chip's two-dimensional topology reduces to a one-dimensional cyclic loop at every program point, and the abstract state of Chapter 8 addresses slug positions by a single integer index $x \in \mathbb{Z}/L\mathbb{Z}$.

The active-path length L depends on the cross-tile mode vector C (7.8), the mixer state s_{mix}, and the I/O mode s_{IO}. With the mixer idle and the I/O zone in ID, the length reduces to a sum over tile contributions,

$$L(C) \;=\; L_{\text{bus}} \;+\; n_{\text{XC}}(C) \cdot \Delta, \qquad n_{\text{XC}}(C) := \sum_{i=0}^{N-1} \mathbb{1}[c_i = \mathsf{XC}], \tag{7.10}$$

where L_{bus} is the all-Bypass length (Section 7.7.1) and Δ the mode-switch shift (Section 7.7.2); MD-mode tiles drop out of the bus path entirely. Path-length increments under non-idle mixer or non-ID I/O modes are bounded by component geometry (Sections 7.6.2, 7.6.3) and are constants of the configuration, not of the program.

7.7.1 Path Composition

In the default *all-Bypass configuration* — every tile in BP, mixer idle, I/O zone idle — with the canonical clockwise ordering pump $\to T_0 \to \cdots \to T_9 \to$ mixer $\to$ I/O $\to$ pump, the active path is the concatenation in Table 7.13.

The total all-Bypass path length is $L_{\text{bus}} = 619$ cells, fixed by the loop circumference $C_{\text{loop}} = 123.8$ mm derived in Section 7.8. The inter-tile gap is the residual: L_{bus} minus the tile, pump, mixer, and I/O

Table 7.13: Active-path composition in the default all-Bypass configuration.

Segment	Count	Cells per instance	Total cells
Tile (BP: $J_U + V_3 + J_D$)	10	3	30
Inter-tile gap	9	≈ 59 (11.9 mm)	534
Pump arc	1	25 (5 mm)	25
Mixing contour (idle: contour sealed)	1	10 (2 mm)	10
I/O zone (idle: bus through)	1	20 (4 mm)	20
		Total L_{bus}	**619**

contributions, distributed across the nine gaps between consecutive tiles.

7.7.2 Mode-Switch Shift

When a single tile transitions from BP to XC, the active-path length increases by

$$\Delta = V_{\text{reg}} + 1 = 417 \text{ cells}, \tag{7.11}$$

the addition of V_1, the V_{reg}-cell serpentine, and V_2, less the removal of V_3. All on-path slugs downstream of the mode-switched tile shift position by $+\Delta$ on the lengthened path; the XC $\rightarrow$ BP transition shifts them by $-\Delta$. This deterministic shift anchors slugs to the discrete state through every mode change and is central to the split and merge semantics of Chapter 10.

Path-length excursion increases the carrier-oil volume in the loop, so the compliance budget of Chapter 5, Section 5.2.3 bounds the simultaneous Exchange count at

$$n_{\text{XC}}(C) \leq 2, \tag{7.12}$$

with $n_{\text{XC}}(C)$ as defined in (7.10). The bound caps the maximal path length at $L_{\text{bus}} + 2\Delta = 1{,}453$ cells ≈ 290.6 mm, matching the $n_{\text{XC}} = 2$ entry of the compliance table of Chapter 5.

7.8 Loop Geometry, Valve Count, and Footprint

The loop dimensions follow from a feasibility lower bound on the bus radius (set by tile-body packing) plus an additive contribution from the non-register components (pump, mixer, I/O zone) along the bus arc.

7.8.1 Loop Derivation from Tile Bounding Box

Each tile occupies a 5×10 mm bounding box (Section 7.4.2), oriented radially inward with the 5 mm dimension tangential to the bus and the 10 mm dimension extending radially. Tight packing of $N = 10$ tile bodies along the inner edge of the bus ring imposes

$$r_{\text{in,min}} = \frac{N \times 5 \text{ mm}}{2\pi} = 7.96 \text{ mm}, \qquad r_{\text{bus,min}} = r_{\text{in,min}} + 10 \text{ mm} = 17.96 \text{ mm}, \tag{7.13}$$

with corresponding minimum loop circumference $C_{\text{loop,min}} = 2\pi\, r_{\text{bus,min}} = 112.8$ mm that accommodates only tile bodies, with no allowance for non-register components.

Adding the non-register bus arcs — pump (5 mm), mixing contour (2 mm), and I/O zone (4 mm) — the loop must therefore satisfy

$$C_{\text{loop}} \;\geq\; C_{\text{loop,min}} + L_{\text{pump}} + L_{\text{mixer}} + L_{\text{IO}} \;=\; 112.8 + 5 + 2 + 4 \;=\; 123.8 \text{ mm}. \tag{7.14}$$

We adopt $C_{\text{loop}} = 123.8$ mm at the lower bound. The bus is approximately circular with effective radius

$$r_{\text{bus}} = \frac{C_{\text{loop}}}{2\pi} \approx 19.7 \text{ mm}, \qquad r_{\text{in}} = r_{\text{bus}} - 10 = 9.7 \text{ mm}, \tag{7.15}$$

the exact shape being a fabrication detail. The 1.74 mm increase from $r_{\text{bus,min}}$ to r_{bus} corresponds to spreading the ten tiles around a slightly larger ring; the resulting inner-edge slack is what opens the inter-tile bus gaps quantified in Section 7.8.2. Discretized at $\ell_v = 200$ μm, this gives $L_{\text{bus}} = C_{\text{loop}}/\ell_v = 619$ cells, the authoritative all-Bypass bus length used throughout the architecture.

7.8.2 Inter-Register Spacing

Each tile occupies 3 cells ($J_U + V_3 + J_D = 0.6$ mm) of bus arc. With ten tiles, the 11 mm non-register zone (pump + mixer + I/O), and $C_{\text{loop}} = 123.8$ mm, the inter-tile gap along the ring is

$$g = \frac{C_{\text{loop}} - 10 \times 0.6 - 11}{9} = \frac{106.8}{9} \approx 11.9 \text{ mm}, \tag{7.16}$$

corresponding to ≈ 59 cells per inter-tile gap (Section 7.7.1). Projected onto the inner edge of the ring, the corresponding arc gap is $g \cdot r_{\text{in}}/r_{\text{bus}} \approx 11.9 \times 9.7/19.7 \approx 5.9$ mm. Each serpentine occupies 4.3 mm of width (Equation 7.3), leaving $5.9 - 4.3 = 1.6$ mm of clearance between adjacent serpentine edges at the inner radius — adequate for fabrication tolerance and control-layer routing.

7.8.3 Valve Count

Eighty-three Quake working valves on the flow layer, summarized in Table 7.14. All 83 valves are addressed through the 128-output Grover demultiplexer (Section 7.6.4).

Table 7.14: On-chip flow-layer valve count.

Component	Count	Valves each	Subtotal
Register tile	10	7	70
Mixing contour	1	3	3
I/O zone	1	7	7
Peristaltic pump	1	3	3
		Total	**83**

The demultiplexer itself adds approximately 160 control-layer valves ($\sim$ 80 latch cells and $\sim$ 80 gain valves) for a total of $\sim$ 243 on-chip valves. None of the demultiplexer valves are on the flow layer, so they do not contribute to fluid resistance or volume budget.

7.8.4 Chip Footprint

The chip diameter $d_{\text{chip}} \approx 2r_{\text{bus}} + 2 \times 2$ mm ≈ 44 mm fits inside a standard 4-inch (100 mm) PDMS wafer [41] and within the ISO 22916 credit-card format (84×54 mm; Section 7.2), with ≈ 2 mm of edge clearance beyond the bus ring for ports.

The flow-layer area decomposes as follows. The central void of radius $r_{\text{in}} \approx 9.7$ mm has area ~ 300 mm^2 and is available for control-layer routing. The annular flow-layer region between r_{in} and r_{bus} has area $\pi(r_{\text{bus}}^2 - r_{\text{in}}^2) \approx 920$ mm^2, of which the ten register tiles consume 500 mm^2 (54%) and the remainder hosts the bus arc, non-register components, and inter-tile margin.

The control-layer area, totaling $\sim$ 1,600 mm^2 across the central void and the rim outside the bus ring, hosts the $\sim$ 72 mm^2 demultiplexer footprint plus pneumatic routing (Section 7.6.4). All external fluid reservoirs — carrier, waste, reagent, sample collection, and the ten module-side connections — are off-chip sealed vials connected by press-fit tubing through ISO-compliant top-face ports.

Chapter 8

Formal State Model

This chapter formalizes the abstract state on which the rest of the system rests. Every property HyDRA's design relies on — valve safety, coalescence safety, register capacity, volume conservation, identifier integrity, asynchronous-future reservation, PARK collision safety — reduces to a predicate over a small integer state S. The chapter fixes the domains, defines S, states the eight invariants every well-formed program preserves, and bounds the state space tightly enough that all preconditions and invariants are decidable in quantifier-free linear integer arithmetic.

The chapter is purely declarative: it sets up the language in which Chapter 9 expresses operational dynamics and Chapter 10 expresses the surface instruction set.

8.1 Domains and Constants

Table 8.1 lists the constants and domains used throughout the chapter; values are inherited from the architecture (Chapter 7, Table 7.1).

Table 8.1: Constants and domains.

Symbol	Value / Type	Description
N	10	Number of register tiles
V_{reg}	416 cells	Register serpentine capacity
L_{bus}	619 cells	All-Bypass active-path length
Δ	$V_{\text{reg}} + 1 = 417$ cells	Mode-switch shift per tile entering XC
Reg	$\{0, \ldots, N-1\}$	Register tile indices
Id	$\mathbb{N}$	Slug identifiers (fresh on allocation, never reused)
FutureId	$\mathbb{N}$	Future identifiers (fresh on allocation, never reused)
Mode	$\{\mathsf{B}, \mathsf{X}\}$	Abstract tile modes
Valve	$\{0, 1\}$	Boolean valve state (1 open, 0 closed)
Dir	$\{+1, -1\}$	Pump direction
$\mathcal{R}_M(p)$	finite $\subset \mathbb{Z}$	Per-module readout codomain (parameter-dependent)
$\mathcal{V}$	$\mathbb{N} \to (\bigcup_M \bigcup_p \mathcal{R}_M(p)) \cup \{\perp\}$	Readout environment
$\mathcal{M}$	finite enumeration	Externally-attached processing modules
$\mathcal{P}_M$	module-specific	Static configuration parameters of module M

The abstract mode set $\{\mathsf{B}, \mathsf{X}\}$ is a two-element abbreviation of the architectural standard modes BP and XC (Chapter 7, Section 7.5.5). The module mode MD halts bus operation for the duration of a module contract and is excluded from the abstract dynamics: it appears only as an internal state of the MODULE_ACCESS compound (Chapter 9, Section 9.5). The transient configurations AC and RP (Chapter 7, Section 7.5.6) are likewise interior to compound operations and not visible at the abstract level. The abstract tile-mode vector is therefore $C \in \mathit{Mode}^N = \{\mathsf{B}, \mathsf{X}\}^N$.

8.2 Active Path Geometry

For configuration $C \in \{\mathsf{B}, \mathsf{X}\}^N$, the active path is the closed cyclic sequence of cells indexed $0, 1, \ldots, L(C)-1$ clockwise from a fixed origin (the clockwise terminus of the pump arc; index 0). The path length

$$L(C) = L_{\text{bus}} + \Delta \cdot \left|\{i : C[i] = \mathsf{X}\}\right| \tag{8.1}$$

is determined by the tile-mode vector and reproduces the architectural decomposition of Chapter 7, Equation 7.10. Cyclic position is over $Pos(C) = \mathbb{Z}_{L(C)}$.

Each tile T_i contributes a switchable segment of length

$$\text{seg}(i, C) = \begin{cases} 1 & C[i] = \mathsf{B} \\ V_{\text{reg}} + 2 & C[i] = \mathsf{X} \end{cases} \tag{8.2}$$

between its upstream and downstream junction cells, denoted $J_U(i, C)$ and $J_D(i, C)$. Tile junction indices satisfy the recurrence

$$J_D(i, C) = J_U(i, C) + \text{seg}(i, C) + 1, \qquad J_U(i{+}1, C) = J_D(i, C) + g_i + 1, \tag{8.3}$$

with fixed inter-tile gaps g_i set at fabrication ($g_i \approx 59$ cells; Chapter 7, Section 7.8.2).

Tile-internal valve cells, expressed as offsets from $J_U(i, C)$ along the active path, are listed in Table 8.2. Module-side valves V_5–V_8 are off-path in both abstract modes and are not used in the formal model.

Table 8.2: Tile-internal valve-cell offsets relative to $J_U(i, C)$ on the active path.

Mode $C[i]$	Valve	Offset from $J_U(i, C)$	Offset from $J_D(i, C)$
B	$V_3(i, C)$	$+1$	-1
X	$V_1(i, C)$	$+1$	—
X	$V_2(i, C)$	$+(V_{\text{reg}} + 2)$	-1

8.3 State

8.3.1 Slug

A slug is a triple

$$\sigma = (id,\, n,\, \ell), \qquad id \in \text{Id}, \quad n \in [1, V_{\text{reg}}], \quad \ell \in Loc, \tag{8.4}$$

where n is the volume (in pump-stroke units, one cell = γ; bound to $[1, V_{\text{reg}}]$ by invariant B, Section 8.4) and the location domain Loc is a tagged union of two cases:

On-path: $\text{Bus}(x)$**.** $x \in Pos(C)$ is the cyclic index of the leading (downstream) face. The slug occupies the cell range $\{(x - n + 1) \bmod L(C), \ldots, x\}$ on the active path. The trailing face is $a := (x - n + 1) \bmod L(C)$.

Off-path: $\text{Reg}(i, d)$**.** The slug is stored in register i with $d \in [0, V_{\text{reg}} + 1]$ the backward offset of the leading face from $V_2(i, \cdot)$. Indices interpret as in Table 8.3.

Table 8.3: Register-internal offset d and corresponding cell role (in X mode for index reference).

d	Cell role	On-path index when $C[i] = \mathsf{X}$
0	V_2 cell (transient)	$J_D(i, C) - 1$
1	Last serpentine cell (park position)	$J_D(i, C) - 2$
V_{reg}	First serpentine cell	$J_U(i, C) + 2$
$V_{\text{reg}} + 1$	V_1 cell (transient)	$J_U(i, C) + 1$

8.3.2 System State

The system state is

$$S = (\Sigma,\ C,\ \nu,\ F), \tag{8.5}$$

with components:

$\Sigma : \text{Id} \rightharpoonup [1, V_{\text{reg}}] \times \mathit{Loc}$**.** Partial finite map from slug identifiers to volume–location pairs. Write $\sigma \in \Sigma$ to mean $\mathit{id} \in \text{dom}(\Sigma)$ and $\Sigma(\mathit{id}) = (n, \ell)$; let $|\Sigma| := |\text{dom}(\Sigma)|$ denote the live-slug count. Carrier oil fills all non-slug space (invariant T, Section 8.4); register i is empty iff no id has $\Sigma(\mathit{id}).\ell = \text{Reg}(i, \_)$.

$C \in \{\mathsf{B}, \mathsf{X}\}^N$**.** Tile-mode vector.

$\nu \in \mathcal{V}$**.** Readout environment. Each module contract declares a finite codomain $\mathcal{R}_M$ at compile time; readouts bind once and are immutable thereafter ($\nu(r) \neq \bot \Rightarrow \nu'(r) = \nu(r)$ for all subsequent ν').

$F : \text{FutureId} \rightharpoonup (\mathcal{M} \times \mathcal{P}_M \times \mathit{Reg})$**.** Pending-future map, recording for each live future the bound module, parameters, and hosting tile.

The initial state is

$$S_0 = (\emptyset,\ [\mathsf{B}, \ldots, \mathsf{B}],\ \nu_0,\ \emptyset), \qquad \nu_0(r) = \bot \text{ for all } r. \tag{8.6}$$

The chip begins with no slugs, all tiles in Bypass, all readouts unbound, and no pending futures.

8.3.3 SSA Discipline

The slug-functional ISA is single-assignment: every identifier (slug $s \in \text{Id}$ or future $f \in \text{FutureId}$) is bound at most once over the lifetime of any program execution. Track this by partitioning the universe of identifiers relative to a state reached at step t:

$$\text{dom}(X) \ =\ \text{Allocated}_X(t) \setminus \text{Consumed}_X(t), \qquad X \in \{\Sigma, F\}, \tag{8.7}$$

where $\text{Allocated}_X(t)$ is the set of all identifiers ever introduced into X through step t and $\text{Consumed}_X(t)$ the set ever removed. The compiler discipline imposes three conditions:

No resurrection. Once consumed, an identifier never re-enters Allocated_X.

No aliasing. Each fresh identifier introduced at step t satisfies $\mathit{id} \notin \text{Allocated}_X(t{-}1)$.

Branch coherence. Any identifier live at branch entry is handled consistently across every arm (Chapter 10, Section 10.4).

These are statically verified by the compiler over the source ISA program and discharge invariant I (Section 8.4). Concretely, the compiler allocates identifiers from monotone-increasing counters, separate for Id and FutureId.

8.4 Invariants

Eight invariants hold after every completed ISA operation, including every branch arm of a branch node and the post-update coalescence phase of every mode switch (Chapter 9, Section 9.2.7). The first three (V, T, P) are formal restatements of the architectural principles of Chapter 5; the remaining five (D, B, C, I, A) cover separation, capacity, conservation, identifier integrity, and asynchronous-future reservation. Table 8.4 states each as a predicate over $S = (\Sigma, C, \nu, F)$.

Table 8.4: The eight HyDRA invariants. $\sigma_j = (id_j, n_j, \ell_j) \in \Sigma$ ranges over live slugs.

	Predicate	Enforcement
V	$\forall j:\ n_j \in \mathbb{N}_{>0}$	Integer stroke counts; split produces integer fragments
T	Carrier oil fills $Pos(C) \setminus \bigcup_j \mathrm{ext}(\sigma_j)$	Maintained by I/O zone CREATE/DESTROY/FLUSH lowering (Chapter 10); not tracked in Σ
P	$C \in \{\mathsf{B}, \mathsf{X}\}^N$; $L(C)$ as in (8.1); $\sum_i \mathbb{1}[C[i] = \mathsf{X}] \le 2$	Mode-switch dynamics (Chapter 9, Section 9.2); compliance bound (Chapter 7, Equation 7.12)
D	$\forall j \neq k:\ \mathrm{ext}(\sigma_j) \cap \mathrm{ext}(\sigma_k) = \emptyset$	Valve-safety precondition; coalescence rule; per-tile register exclusion
B	$\forall j:\ 1 \le n_j \le V_{\mathrm{reg}}$	create bounded; merge requires $n_1 + n_2 \le V_{\mathrm{reg}}$
C	$\sum_j n_j$ changes only on explicit I/O and module operations	mix, hold, flush, mode switches preserve $\sum n_j$
I	$\mathrm{dom}(\Sigma) = \mathrm{Allocated}_\Sigma \setminus \mathrm{Consumed}_\Sigma$; same for F	SSA discipline (Section 8.3.3)
A	$\forall f \in \mathrm{dom}(F):\ C[F(f).i_M] = \mathsf{B}$ and tile i_M's register is empty; $f \mapsto F(f).i_M$ injective	process_async preconditions; await releases reservation

Slug extent. For an on-path slug $\sigma = (id, n, \mathrm{Bus}(x))$, the cap-extended extent used in invariant D is

$$\mathrm{ext}(\sigma) = \big[(x - n) \bmod L(C),\ (x + 1) \bmod L(C)\big], \tag{8.8}$$

extending the slug's occupied range by one cell on each side to enforce ≥ 1 cell of carrier oil between any two on-path slugs. For an off-path slug $\sigma = (id, n, \mathrm{Reg}(i, d))$, $\mathrm{ext}(\sigma) = \mathrm{Reg}(i, \cdot)$ (the entire register, since at most one slug is permitted per register by D).

Invariant D, formal. For all distinct on-path slugs σ_j, σ_k with $\ell_j = \mathrm{Bus}(x_j)$ and $\ell_k = \mathrm{Bus}(x_k)$:

$$[x_j - n_j,\ x_j + 1] \cap [x_k - n_k,\ x_k + 1] = \emptyset \pmod{L(C)}. \tag{8.9}$$

Reaching zero cap-to-cap separation triggers physical coalescence (Chapter 9, Section 9.2.6); coalescence between slugs that the program did not explicitly merge is an error caught by D. For off-path slugs:

$$\forall j \neq k:\ \neg\big(\ell_j = \mathrm{Reg}(i,\ \_\ )\ \wedge\ \ell_k = \mathrm{Reg}(i,\ \_\ )\big). \tag{8.10}$$

Invariant A, formal. Let $R(F) = \{F(f).i_M : f \in \mathrm{dom}(F)\}$. Then

$$\forall i \in R(F):\ C[i] = \mathsf{B}\ \wedge\ \neg\exists\, id \in \mathrm{dom}(\Sigma):\ \Sigma(id).\ell = \mathrm{Reg}(i,\ \_\ ), \tag{8.11}$$

and $f \mapsto F(f).i_M$ is injective, so $|R(F)| = |\mathrm{dom}(F)| \le N$.

Termination obligation. At program exit, $\mathrm{dom}(F) = \emptyset$: no future is left unawaited. The compiler enforces this statically as part of SSA discipline.

8.5 State Space Bounds

Each component of S is bounded as follows.

Configurations. Under invariant P's compliance bound $n_{\mathrm{XC}}(C) \le 2$,

$$|\mathcal{C}_{\mathrm{reach}}| = \binom{N}{0} + \binom{N}{1} + \binom{N}{2} = 56. \tag{8.12}$$

Live slugs. Each live slug occupies ≥ 1 cell with ≥ 1 cell of carrier separation by D, so

$$|\Sigma| \;\le\; N + \left\lfloor \frac{L_{\mathrm{bus}} - K_{\mathrm{sep}}}{2} \right\rfloor, \tag{8.13}$$

where the first term is the register file (one slug per register) and K_{sep} collects bus arc cells unavailable for slug residence. For HyDRA's reference layout, K_{sep} comprises the pump arc (25 cells), the mixer entry/exit (10 cells), and the I/O zone (20 cells), totalling $K_{\mathrm{sep}} = 55$ cells; substituting gives $|\Sigma| \le 10 + \lfloor (619 - 55)/2 \rfloor = 10 + 282 = 292$. For typical assays with slug volumes 5–50 cells, the binding constraint is the register file rather than this bus-residence bound.

Pending futures. By A, $|\mathrm{dom}(F)| \le N$.

Volume. $\sum_j n_j \le N \cdot V_{\mathrm{reg}} + L_{\mathrm{bus}} - K_{\mathrm{sep}}$.

Identifiers. Id and FutureId are unbounded as types, but a program of m operations and branch depth b allocates at most $O(m \cdot (q{+}1)^b)$ identifiers — finite by program structure. The state S is finite up to identifier renaming.

8.6 Tractability

The HyDRA verification problem fits inside quantifier-free linear integer arithmetic.

QF-LIA fragment. Every precondition and every invariant in this chapter is a Boolean combination of linear inequalities

$$\sum a_i x_i \le c, \qquad a_i, c \in \mathbb{Z}, \tag{8.14}$$

over the integer state components. Readouts $\nu(r)$ appear as free integer variables with finite codomains $\mathcal{R}_M(p)$, eliminable by enumeration. Branch predicates (Chapter 10, Section 10.4) are constructed from the same fragment. There are no quantifiers and no products of state variables.

Decidability. Quantifier-free linear integer arithmetic is decidable as a sub-fragment of Presburger arithmetic, and the state-space bounds of Section 8.5 make the search finite. The verification algorithm of Chapter 9, Section 9.7 discharges all checks by a forward symbolic walk over the lowered program with cost

$$O\big(m \cdot (q{+}1)^b \cdot |\Sigma|_{\max}^2\big), \tag{8.15}$$

where m is program length, b branch depth, q maximum branch arity, and $|\Sigma|_{\max}$ the maximum simultaneously-live slug count. The pairwise adjacency check for invariant D dominates. For the serial-dilution example (Chapter 10, $m \approx 30$, $|\Sigma|_{\max} = 5$, $b = 0$), this is ≈ 750 integer comparisons; published continuous-flow benchmarks ($m \le 200$, $|\Sigma|_{\max} \le 10$, $b \le 3$, $q \le 2$) sit below 6×10^5 comparisons — negligible compile-time overhead.

No prior programmable-microfluidic architecture achieves equivalent decidability: continuous-state sys-

tems (BioStream [40], AquaCore [3]) cannot reduce their state space to integer inequalities; EWOD-based systems (BioScript [31], Puddle [45]) check chemistry-level safety only or defer correctness to runtime.

8.7 Physical Bridge

The formal model treats positions and volumes as exact integers; the physical substrate is continuous. Soundness of integer reasoning over the continuous substrate rests on the architectural principles of Chapter 5. Volume is quantized by V (every create produces an integer-cell slug; every split produces integer fragments). Position is quantized by T (the cell grid maps to physical channel segments) up to the drift bound $|\Delta x(n)| \leq 0.04\,n$ cells, which is zeroed at every PARK checkpoint (Chapter 9, Section 9.4). Active-path topology is constrained by P: C alone determines $L(C)$, so the abstract-physical correspondence depends only on C being well-formed and the path being closed.

Together, these guarantee that $S = (\Sigma, C, \nu, F)$ accurately models the physical chip at every PARK checkpoint, which Chapter 9 establishes as the boundary of every compound microarchitectural operation. The full soundness theorem — that any execution trace of a verified program preserves all eight invariants and satisfies every precondition — is stated in Chapter 9, Section 9.8, after the operational dynamics that connect S to the primitive operations are defined.

Chapter 9

Microarchitecture

Between the chip layout of Chapter 7 and the instruction set of Chapter 10 sits a fixed layer of five primitive operations. Every compound ISA operation lowers deterministically into a finite sequence over these five. This chapter specifies the primitives, the operational semantics of tile-mode transitions, the position- and volume-error budget and the PARK procedure that bounds it, the compound microarchitectural operations the compiler builds upon, and the scheduling discipline that emits the timed primitive stream.

9.1 Primitive Operations

HyDRA exposes five primitive operations: CIRC (pump actuation), SETR (register-tile configuration), SETMX (mixer junction), SETIO (I/O zone configuration), and WAIT (timed hold). Each has a fixed signature, a deterministic effect on the chip, and a deterministic duration. Splitting, coalescence, parking, and transport are not primitives; they are physical consequences of specific primitive sequences acting on the valve configuration and fluid regime. Table 9.1 summarizes; the rest of this section gives each primitive in turn.

Table 9.1: Microarchitectural primitives.

Primitive	Signature	Effect	Duration
CIRC	(n, v, d)	Advance on-path slugs n cells at velocity v, direction d	nt_s or nt_p
SETR	(i, M)	Set tile T_i to mode $M \in \{\mathsf{BP}, \mathsf{XC}, \mathsf{MD}, \mathsf{AC}, \mathsf{RP}\}$	$H_i \cdot t_v$
SETMX	(s)	Set mixer to state $s \in \{0, 1\}$ (isolated/active)	$H_{\text{mix}} \cdot t_v$
SETIO	(M)	Set I/O zone to mode $M \in \{\mathsf{ID}, \mathsf{IN}, \mathsf{FL}, \mathsf{EJ}\}$	$H_{\text{IO}} \cdot t_v$
WAIT	(t)	Hold valve and pump state for t ms	t ms

The four SET* primitives reconfigure valves with the pump idle — a consequence of principle T (Chapter 5): changing active-path length while fluid is in motion would break the stroke-to-cell correspondence. CIRC is the only primitive that advances fluid; WAIT advances no fluid and changes no valve.

9.1.1 CIRC

$$\textsc{Circ}(n, v, d), \qquad n \in \mathbb{Z}^+, \quad v \in \{v_{\text{nom}}, v_{\text{prec}}\}, \quad d \in \{+, -\}. \tag{9.1}$$

Actuate the peristaltic pump for n strokes at velocity v in direction d. The transition rule on the state $S = (\Sigma, C, \nu, F)$ updates only the location of on-path slugs:

$$\sigma = (id, n_\sigma, \text{Bus}(x)) \xrightarrow{\textsc{Circ}(n,v,+)} (id, n_\sigma, \text{Bus}((x + n) \bmod L(C))), \tag{9.2}$$

with C, ν, F, identifiers, and off-path slugs ($\mathrm{Reg}(i,d)$) unchanged. Reverse pumping ($d=-$) shifts by $-n$ and is used only by EJECT (Section 9.5). CIRC performs no valve change; the two velocities and two directions are summarized in Table 9.2.

Table 9.2: CIRC parameters.

Parameter	Value	Role
v_{nom}	10 mm/s (50 Hz, $t_s = 20$ ms/stroke)	Transport (Chapter 7)
v_{prec}	1 mm/s (5 Hz, $t_p = 200$ ms/stroke)	Precision approach in PARK Phase 3 (Section 9.4)
$d=+$	Clockwise: pump $\to T_0 \to \cdots \to T_9 \to$ I/O $\to$ pump	Default forward direction
$d=-$	Counter-clockwise	Used only by EJECT (Section 9.5)

The duration is $n \cdot t_s$ for $v = v_{\mathrm{nom}}$ and $n \cdot t_p$ for $v = v_{\mathrm{prec}}$. A CIRC that immediately precedes a tile reconfiguration is constrained by the valve-safety precondition (Section 9.2.4): on completion, no slug face may sit on a valve cell that is about to close.

9.1.2 SETR

$$\textsc{SetR}(i, M), \qquad i \in [0, N-1], \quad M \in \{\mathsf{BP}, \mathsf{XC}, \mathsf{MD}, \mathsf{AC}, \mathsf{RP}\}. \tag{9.3}$$

Set tile T_i to valve configuration M (modes defined in Chapter 7, Sections 7.5.5–7.5.6). The Grover demultiplexer addresses one valve per cycle, so reconfiguration time scales with the Hamming distance between current and target valve states:

$$t_{\textsc{SetR}}(i, M_{\mathrm{old}} \to M_{\mathrm{new}}) = H_i(M_{\mathrm{old}}, M_{\mathrm{new}}) \cdot t_v, \tag{9.4}$$

where H_i is the number of valves whose state changes and $t_v = 120$ ms is the per-valve set time (Chapter 7, Section 7.6.4). For BP $\leftrightarrow$ XC, $H_i = 3$ (V_1, V_2, V_3); for BP $\leftrightarrow$ MD, $H_i = 7$ (every tile valve toggles).

The architectural mode set $\{\mathsf{BP}, \mathsf{XC}, \mathsf{MD}, \mathsf{AC}, \mathsf{RP}\}$ that SETR targets relates to the abstract mode set $\{\mathsf{B}, \mathsf{X}\}$ of Chapter 8 as follows: $\mathsf{B} \equiv \mathsf{BP}$ and $\mathsf{X} \equiv \mathsf{XC}$ at the abstract level; MD is a session boundary that freezes the abstract dynamics for the duration of a module contract; AC and RP are transient configurations interior to compound operations (Section 9.5) and never appear at the abstract level.

9.1.3 SETMX, SETIO

$$\textsc{SetMx}(s), \qquad s \in \{0, 1\}; \qquad \textsc{SetIO}(M), \qquad M \in \{\mathsf{ID}, \mathsf{IN}, \mathsf{FL}, \mathsf{EJ}\}. \tag{9.5}$$

SETMX sets the mixing-contour T-junction (Chapter 7, Section 7.6.2): $s = 0$ isolates the contour (default; bus through bypass), $s = 1$ routes the active path through the contour. Three valves toggle; duration is $\leq 3t_v = 360$ ms.

SETIO sets the I/O zone (Chapter 7, Section 7.6.3). Valve state 1 open, 0 closed (matching the convention of Chapter 7). Table 9.3 gives the per-mode valve assignments.

Table 9.3: SETIO mode encoding. Valve state 1 (open) and 0 (closed). V_A, V_B, V_C are bus-segment valves; $V_{\mathrm{wo}}, V_{\mathrm{in}}, V_{\mathrm{ci}}, V_{\mathrm{ro}}$ are stub valves (Chapter 7, Equation 7.9).

Mode	V_A	V_B	V_C	V_{wo}	V_{in}	V_{ci}	V_{ro}	Pump	Effect per stroke
ID	1	1	1	0	0	0	0	idle	Bus through, all stubs sealed
IN	0	1	1	1	1	0	0	+	γ reagent in, γ carrier out to waste
FL	1	0	1	1	0	1	0	+	γ fresh carrier in, γ old fluid out to waste
EJ	1	1	0	0	0	1	1	−	γ carrier in, γ slug material out to collection

Duration is $H_{\mathrm{IO}} \cdot t_v$, where H_{IO} is the Hamming distance between the current and target seven-bit valve states. ID $\leftrightarrow$ active transitions toggle three valves ($\sim$ 360 ms); active $\leftrightarrow$ active transitions toggle four to six.

9.1.4 WAIT

$$\textsc{Wait}(t), \qquad t \in \mathbb{Z}^{+} \text{ ms.} \tag{9.6}$$

Hold the current valve configuration and pump state for t ms with no fluid movement and no valve change. Used for coalescence dwell ($\sim t_c \approx 10$ s; Chapter 6), thermal equilibration, incubation, and any timed hold at a fixed configuration.

9.2 Tile Mode Transitions

A mode change at tile T_i is realized by a single SetR, but its effect on the abstract state Σ depends on the direction, the positions of any on-path slugs relative to the tile, and the cells whose valves close during the transition. This section gives the formal transition rules for each direction, the valve-safety precondition that rules out ambiguous splits, the post-transition coalescence rule, and the execution order in which the rules are applied. The architectural valve-toggle sequences underlying SetR are in Chapter 7, Section 7.5.8.

9.2.1 Closing-Valve Set

For a transition $M_{\mathrm{old}} \to M_{\mathrm{new}}$ at tile T_i, the set of valve cells that close during the transition is

$$\mathrm{clos}(i, M_{\mathrm{old}} \to M_{\mathrm{new}}) = \begin{cases} \{V_3(i,C)\} & \mathsf{B} \to \mathsf{X}, \\ \{V_1(i,C),\, V_2(i,C)\} & \mathsf{X} \to \mathsf{B}, \\ \{V_3(i,C),\, V_5(i,C),\, V_6(i,C)\} & \mathsf{B} \to \mathsf{MD}, \\ \{V_1(i,C),\, V_2(i,C),\, V_7(i,C),\, V_8(i,C)\} & \mathsf{MD} \to \mathsf{B}. \end{cases} \tag{9.7}$$

Module-side valve cells (V_5–V_8) are off-path in both abstract modes (Chapter 8, Section 8.2), so only the $\mathsf{B} \leftrightarrow \mathsf{X}$ entries enter the abstract dynamics of Sections 9.2.2 and 9.2.3. $\mathsf{B} \leftrightarrow \mathsf{MD}$ transitions are session boundaries: they freeze on-path slug positions for the duration of the module contract (the bus pump is held idle while the bus loop is broken at T_i) and bracket the MODULE_ACCESS compound of Section 9.5.

9.2.2 Bypass to Exchange

Let $C' = C[i \mapsto \mathsf{X}]$ with $C[i] = \mathsf{B}$. For each $\sigma = (id, n_\sigma, \mathrm{Bus}(x)) \in \Sigma$ with trailing face $a = x - n_\sigma + 1$, Table 9.4 gives the position update. Off-path slugs at $\mathrm{Reg}(i,d)$ rejoin the active path inside the now-engaged serpentine; off-path slugs at $\mathrm{Reg}(j,d)$ for $j \neq i$ are unaffected.

The Spanning rule produces two fragments: an upstream fragment of k cells crossing into the newly-engaged serpentine and a downstream fragment of $n_\sigma - k$ cells at the upstream junction. Volume is conserved: the fragment volumes sum to n_σ. The compiler rules out the Spanning case at compile time via the valve-safety precondition (Section 9.2.4); the rule is included for completeness of the dynamics.

9.2.3 Exchange to Bypass

Let $C' = C[i \mapsto \mathsf{B}]$ with $C[i] = \mathsf{X}$. Off-path slugs at $\mathrm{Reg}(j,d)$ for any j are unaffected. For each $\sigma = (id, n_\sigma, \mathrm{Bus}(x)) \in \Sigma$, Table 9.5 gives the position update.

The Inside-serpentine case relocates an in-bus slug back into the off-path register, with the leading-face offset $d = J_D(i,C) - 1 - x$ chosen so that the slug occupies the same physical cells before and after. As

Table 9.4: $\mathsf{B} \to \mathsf{X}$ transition at tile T_i. Identifiers are preserved unless the Spanning case fires, in which case id is consumed and two fresh identifiers id_a, id_b are allocated by the lowering site.

Case	Condition	Update
Upstream of tile	$x < V_3(i, C)$	Σ unchanged for σ
Downstream of tile	$a \geq J_D(i, C)$	$\Sigma(id) \leftarrow (n_\sigma,\ \mathrm{Bus}(x + \Delta))$
Spanning V_3	$x > V_3(i, C) \wedge n_\sigma > k$, where $k = x - V_3(i, C)$	$\Sigma \leftarrow \Sigma \setminus \{id\}$, then $\Sigma(id_a) \leftarrow (k,\ \mathrm{Bus}(J_D(i, C') + k - 1))$, $\Sigma(id_b) \leftarrow (n_\sigma - k,\ \mathrm{Bus}(J_U(i, C')))$
Off-tile register	$\ell = \mathrm{Reg}(j, d)$, $j \neq i$	Σ unchanged for σ
On-tile register	$\ell = \mathrm{Reg}(i, d)$	$\Sigma(id) \leftarrow (n_\sigma,\ \mathrm{Bus}(J_D(i, C') - 1 - d))$

Table 9.5: $\mathsf{X} \to \mathsf{B}$ transition at tile T_i. Identifiers preserved unless a Spanning case fires; allocation as in Table 9.4.

Case	Condition	Update
Upstream of tile	$x \leq J_U(i, C)$	Σ unchanged for σ
Inside serpentine	$a > V_1(i, C) \wedge x < V_2(i, C)$	$\Sigma(id) \leftarrow (n_\sigma,\ \mathrm{Reg}(i,\ J_D(i, C) - 1 - x))$
Downstream of tile	$a \geq J_D(i, C)$	$\Sigma(id) \leftarrow (n_\sigma,\ \mathrm{Bus}(x - \Delta))$
Spanning V_1	$x > V_1(i, C) \wedge n_\sigma > k$, where $k = x - V_1(i, C)$	$\Sigma \leftarrow \Sigma \setminus \{id\}$, then $\Sigma(id_a) \leftarrow (k,\ \mathrm{Reg}(i,\ V_{\mathrm{reg}} - k + 1))$, $\Sigma(id_b) \leftarrow (n_\sigma - k,\ \mathrm{Bus}(J_U(i, C')))$
Spanning V_2	$x > V_2(i, C) \wedge n_\sigma > k$, where $k = x - V_2(i, C)$	$\Sigma \leftarrow \Sigma \setminus \{id\}$, then $\Sigma(id_a) \leftarrow (k,\ \mathrm{Bus}(x - \Delta))$, $\Sigma(id_b) \leftarrow (n_\sigma - k,\ \mathrm{Reg}(i,\ 0))$

with $\mathsf{B} \to \mathsf{X}$, total volume is conserved across all cases.

9.2.4 Valve-Safety Precondition

A closing valve that intersects a slug body would split the slug at an unknown position; a closing valve at a slug cap would ambiguate the cap position. Both are architectural errors and rule out cleanly-defined transitions. The compiler enforces, for every on-path slug $\sigma = (id, n_\sigma, \mathrm{Bus}(x)) \in \Sigma$ with trailing face $a = x - n_\sigma + 1$:

$$x \notin \mathrm{clos}(i, M_{\mathrm{old}} \to M_{\mathrm{new}}) \ \wedge \ a \notin \mathrm{clos}(i, M_{\mathrm{old}} \to M_{\mathrm{new}}). \tag{9.8}$$

A slug may *span* a closing valve (triggering a Spanning case in Section 9.2.2 or 9.2.3), but neither cap may rest on the closing valve cell. The precondition is a Boolean combination of integer linear inequalities over Σ and the lithographically-known valve indices, statically discharged because every compound operation begins at a PARK checkpoint where all slug positions are exactly known (Section 9.4).

9.2.5 Split Tolerance

When a valve closes on a cell containing slug material (a Spanning case in Section 9.2.2 or 9.2.3), the descending membrane distributes the cell's contents to both sides in a physically uncontrolled ratio. The formal model assigns the valve cell to the upstream fragment by convention, accepting a ± 1 cell uncertainty per split. This uncertainty is comparable to the per-stroke pump deviation ε (Section 9.3) and is absorbed by the same tolerance budget; routing safety is verified over nominal integers, while assay accuracy carries $\pm\gamma$ per split. Legitimate compilation never schedules a transition with a Spanning slug — the compiler PARKs slugs clear of every closing valve before each mode switch — so the Spanning

rules are dynamics fallbacks rather than user-visible operations.

9.2.6 Post-Transition Coalescence

After all position updates of Section 9.2.2 or 9.2.3, two on-path slugs may end up adjacent on the new active path — separated by zero cells of carrier oil. This occurs when an opening valve in the transition removes the physical barrier that previously separated them. Coalescence then proceeds spontaneously (Chapter 6, Section 6.5.3).

Adjacency. Two on-path slugs $\sigma_1 = (id_1, n_1, \mathrm{Bus}(x_1))$ and $\sigma_2 = (id_2, n_2, \mathrm{Bus}(x_2))$ are adjacent under C' iff

$$(x_1 + 1) \equiv (x_2 - n_2 + 1) \pmod{L(C')} \quad \vee \quad (x_2 + 1) \equiv (x_1 - n_1 + 1) \pmod{L(C')}. \tag{9.9}$$

Coalescence rule. If adjacent (with σ_1 upstream of σ_2, taking the first disjunct of (9.9)), they merge into a single fresh-id slug:

$$\Sigma \leftarrow \big(\Sigma \setminus \{id_1, id_2\}\big) \cup \big\{id_3 \mapsto (n_1 + n_2,\, \mathrm{Bus}(x_2))\big\}, \tag{9.10}$$

with id_3 allocated fresh at the lowering site. Coalescence requires $n_1 + n_2 \le V_{\mathrm{reg}}$ (invariant B); the compiler rejects programs that produce an over-capacity coalescence at compile time.

This is the formal fallback for any operation sequence that produces unintended adjacency. It is *not* the mechanism by which the ISA's merge executes: merge is defined atomically at the ISA level (Chapter 10) and lowers via a dedicated MERGE compound (Section 9.5) that drives both slugs against opposite faces of a closed V_2, opens V_2, and coalesces through equilibrium-film contact at sub-saturation surfactant coverage (Chapter 6). The post-coalescence geometry of merge does not pass through the cell-level adjacency predicate (9.9); the formal model treats the ISA-level transition as atomic. Equation (9.10) covers only residual unintended adjacency, which invariant D rules out.

9.2.7 Execution Order

Every mode switch proceeds in three phases:

1. **Pre-check.** Verify the valve-safety precondition (9.8).
2. **Update.** Apply the position-update rules of Section 9.2.2 or 9.2.3, including any Spanning splits and any Reg $\leftrightarrow$ Bus relocations.
3. **Coalesce.** Scan for adjacent on-path slug pairs and apply the coalescence rule (9.10) where it fires.

Invariants V/T/P/D/B/C/I/A (Chapter 8, Section 8.4) are checked after Phase 3. The valve-safety precondition is the only Phase-1 check; Phase 2 produces deterministic state given a satisfied precondition; Phase 3 is the only phase that may fire on σ pairs neither of which was a Phase-2 source.

9.2.8 Fluid Effects of Mode Switching

Mode switching occurs with the pump idle, so no *net* fluid advance is induced by the transition itself. Two minor effects are absorbed locally without violating invariant T or V.

Volume redistribution. A B $\leftrightarrow$ X transition changes the active-path length by $\Delta = 417$ cells but not the total fluid volume in the loop. The serpentine is full of carrier oil at all times (invariant T), so opening V_1, V_2 merely connects an already-full segment to the active path; closing them disconnects it. No fluid is created, compressed, or expelled by the transition.

Valve membrane displacement. When a Quake valve closes, the deflecting membrane displaces $\sim \gamma/2$ from its cell into adjacent corner gutters; the reverse occurs on opening. These sub-cell, bounded displacements are local to the valve cell and do not affect slug positions at the cell-grid level. They contribute to the per-stroke pump-quantum variance analyzed in Chapter 5, Section 5.1.2.

9.3 Error Management

The architecture counts strokes and cells as integers; physical reality has per-stroke fluctuations, continuous slug motion, and per-chip fabrication tolerances. This section formalizes the gap and bounds it.

9.3.1 Error Sources

Three physical deviations contribute to the abstract-physical gap (Table 9.6). After per-chip calibration, only the first two accumulate during operation.

Table 9.6: Physical sources of position and volume deviation.

Source	Bound	Accumulates in	Reset by
Slug drift, α	$\lvert\alpha\rvert \leq 0.03$	Position (per stroke)	PARK
Per-stroke deviation, ε_k	$\lvert\varepsilon_k\rvert \leq 0.01$	Position and volume (per stroke)	PARK (position only)
Systematic offset, $\gamma_{\text{actual}}/\gamma$	$\in [0.68, 1.33]$	Absolute volume per stroke	Per-chip calibration scalar

Slug drift. The Aussillous–Quéré expression (Chapter 6, Section 6.2) gives a steady-state slug velocity $(1+\alpha)$ times the mean carrier velocity, where $|\alpha| \leq 0.03$ at the operating capillary number. The drift is systematic: it accumulates with the same sign on every stroke during a transit.

Per-stroke deviation. The Goulpeau pump produces sub-1% stroke-to-stroke variation in the linear regime (Chapter 5, Section 5.1.2): the actual displacement on stroke k is $\gamma_{\text{actual}}(1+\varepsilon_k)$ with $|\varepsilon_k| \leq 0.01$. This is stochastic: the sign of ε_k varies stroke to stroke.

Systematic offset. The per-chip ratio $\gamma_{\text{actual}}/\gamma$ ranges from 0.68 to 1.33 across fabrication tolerances. It is absorbed at commissioning by the per-chip calibration scalar $s = \gamma/\gamma_{\text{actual}}$ applied at cell-to-stroke translation, so each issued ISA stroke physically delivers the assumed γ of fluid. After calibration, the worst-case accumulated error over n ISA strokes is $n\,\varepsilon_{\max}$ cells regardless of s. The systematic offset is not considered further in the error model below.

9.3.2 Combined Per-Stroke Advance

A single ISA stroke advances a slug by $(1+\alpha)(1+\varepsilon_k)$ cells rather than exactly one. The worst-case bounds govern all subsequent error analysis:

$$a_{\max} := (1+\alpha_{\max})(1+\varepsilon_{\max}) = 1.03 \times 1.01 = 1.0403, \tag{9.11}$$

$$a_{\min} := (1-\alpha_{\max})(1-\varepsilon_{\max}) = 0.97 \times 0.99 = 0.9603. \tag{9.12}$$

These two constants appear throughout the PARK analysis of Section 9.4.

9.3.3 Position Uncertainty

After n strokes of Circ without re-registration, the physical slug position can deviate from the abstract (stroke-counted) position by

$$|\Delta x(n)| \le n \cdot \max(a_{\max} - 1,\ 1 - a_{\min}) = 0.0403\, n \ \le\ 0.04\, n \text{ cells.} \tag{9.13}$$

The bound is dominated by the α contribution: drift accumulates with consistent sign every stroke, while ε_k fluctuations partially cancel in expectation but do not in the worst case. Representative values are tabulated in Table 9.7.

Table 9.7: Position uncertainty after n strokes without re-registration.

n (strokes)	Context	Worst-case drift (cells)	Physical drift
10	Short transfer	0.4	80 μm
60	Adjacent-tile transit	2.4	480 μm
310	Half-loop transit	12.5	2.5 mm
619	Full-loop transit	24.9	5.0 mm

For $n \gtrsim 25$ strokes, the drift exceeds one cell and any operation that depends on cell-precise positioning must be preceded by re-registration (Section 9.4).

9.3.4 Volume Uncertainty

When a slug is created or transferred by k strokes of CIRC, its actual volume is

$$V_{\text{actual}} \ =\ \gamma_{\text{actual}} \sum_{i=1}^{k} (1 + \varepsilon_i) \ =\ k\, \gamma_{\text{actual}}\, (1 + \bar{\varepsilon}), \tag{9.14}$$

where $\bar{\varepsilon} = (1/k) \sum \varepsilon_i$ is the empirical mean fluctuation. The worst-case error is

$$|V_{\text{actual}} - k\gamma_{\text{actual}}| \ \le\ k\, \varepsilon_{\max}\, \gamma_{\text{actual}}. \tag{9.15}$$

Volume error is fixed at slug creation: subsequent CIRC operations are volume-conserving in steady state, since the Bretherton film deposited at the trailing cap is balanced by the film already present along the channel ahead of the leading cap (Chapter 6). Corner-gutter mass transfer is rate-limited by the negligible aqueous solubility in HFE-7500 and is bounded over assay timescales (Chapter 6, Section 6.5.2). There is no analogue of re-registration for volume; $k\, \varepsilon_{\max}$ is the architecture's intrinsic volume precision. After merge, input uncertainties add: $|n_3 - (n_1 + n_2)| \le |n_1 - \bar{n}_1| + |n_2 - \bar{n}_2|$. Representative values are in Table 9.8.

Table 9.8: Worst-case volume error for k-stroke slug creation.

k (strokes)	Nominal volume	Worst-case error	Relative
10	6.0 nL	$\pm$0.06 nL	$\pm$1.0%
50	30.0 nL	$\pm$0.30 nL	$\pm$1.0%
$V_{\text{reg}} = 416$	250.0 nL	$\pm$2.50 nL	$\pm$1.0%

9.3.5 Error Discharge

Position uncertainty is the only accumulating error of architectural concern after calibration; PARK (Section 9.4) zeroes it. Volume uncertainty is fixed at creation and cannot be reset by any architectural mechanism: it sets the absolute precision floor of any assay, and the compiler propagates it forward as a static bound on every slug.

9.4 Re-Registration (PARK)

PARK is the re-registration procedure that zeroes accumulated positional drift. It drives a slug cap into physical contact with a closed valve membrane, after which the slug's position is exactly known: the cap is at the lithographic position of the valve. PARK is not a primitive but a deterministic composition of three phases.

Phase 1 (Transport). CIRC$(n_{\text{stop}}, v_{\text{nom}}, +)$ advances the slug toward the target valve, which remains open during the phase. The stroke count

$$n_{\text{stop}} = \left\lfloor \frac{d - \delta_{\text{safe}}}{a_{\max}} \right\rfloor \tag{9.16}$$

is chosen so that even in worst-case over-advance ($a_{\max}$ per stroke), the slug cap remains $\geq \delta_{\text{safe}}$ cells upstream of the target. Here d is the cap-to-valve distance at PARK start and $\delta_{\text{safe}} \geq 1$ cell is the minimum carrier-oil gap.

Phase 2 (Valve close). SETR(i, M) closes the target valve onto the carrier oil between the slug and the valve face. The Phase 1 bound $\text{gap}_{\min} \geq \delta_{\text{safe}}$ guarantees no slug material is caught in the closing valve cell, excluding Gilet's regime S [14].

Phase 3 (Precision approach). CIRC$(n_{\text{TF}}, v_{\text{prec}}, +)$ at $v_{\text{prec}} = v_{\text{nom}}/10$ drops the capillary number to $Ca = 7.3 \times 10^{-5}$, placing the slug in Gilet's TF regime where it is pressed against the closed valve face without splitting or escaping through corner gutters. The stroke count

$$n_{\text{TF}} = \left\lceil \frac{\text{gap}_{\max}}{a_{\min}} \right\rceil, \qquad \text{gap}_{\max} = d - n_{\text{stop}}\, a_{\min}, \tag{9.17}$$

is chosen so that even in worst-case under-advance ($a_{\min}$ per stroke), the slug cap reaches the valve face.

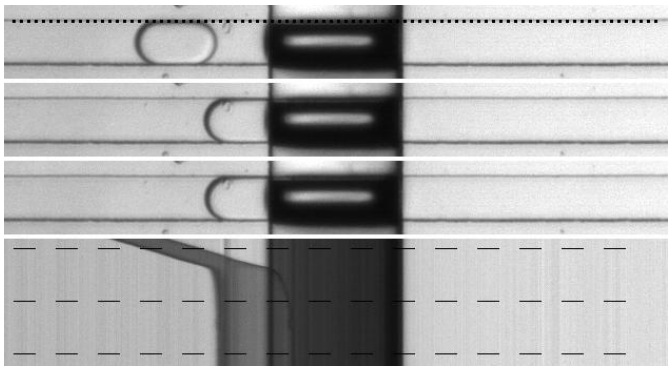


Figure 9.1: The TF (trapped-in-front) regime exploited by PARK Phase 3, from Gilet and van Loo's experimental classification of slug–valve interactions [14]. The three image strips at top show successive snapshots (separated by 0.4 s) of an aqueous slug approaching a closed pneumatic valve (dark vertical bar) at low capillary number; the slug presses flush against the closed valve face and remains there indefinitely. The lower panel is a spatio-temporal diagram (horizontal axis: position along channel; vertical axis: time) showing the slug at rest while the carrier oil continues to flow around it through the channel-corner gutters. HyDRA's PARK Phase 3 operates in this regime by design. Reproduced from Gilet and van Loo [14].

9.4.1 Safety Properties

Regime S exclusion. After Phase 1, the minimum remaining gap satisfies

$$\text{gap}_{\min} = d - n_{\text{stop}} \cdot a_{\max} \;\geq\; \delta_{\text{safe}} \;>\; 0 \tag{9.18}$$

by construction of n_{stop}, so the valve closes onto carrier oil and no slug material is caught in the closing valve cell.

Contact guarantee. Phase 3 delivers n_{TF} strokes, each advancing the slug by at least $a_{\min}$ cells; by construction $n_{\mathrm{TF}} \cdot a_{\min} \geq \mathrm{gap}_{\max}$, so contact is reached even in worst-case under-advance.

Invariants preserved. PARK uses only CIRC and SETR. No primitive in this set creates, destroys, or partitions slug volume, so invariants V (volume quantization), T (always-full loop), P (single active path), and D (inter-slug separation) are all preserved.

9.4.2 Timing

Table 9.9: PARK timing for representative cap-to-valve distances d. Stroke periods: $t_s = 20$ ms at v_{nom} (Phase 1), $t_p = 200$ ms at v_{prec} (Phase 3); plus $t_v = 120$ ms for the Phase 2 valve cascade.

d (cells)	Context	n_{stop}	n_{TF}	Total strokes	Wall time
30	half-tile-gap	27	5	32	1.66 s
60	adjacent-tile transit	56	7	63	2.64 s
124	two-tile transit	118	12	130	4.88 s
310	half-loop transit	297	26	323	11.26 s

PARK overhead is additive for every non-local compound operation (Section 9.5). Long-distance operations are dominated by PARK time, not valve switching. The scheduler's first optimization is therefore to minimize the PARK target distance by choosing the nearest valid anchor valve; valve-cascade and pipelining optimizations are secondary (Section 9.6).

9.4.3 Collision Safety

All on-path slugs advance by $n_{\mathrm{stop}} + n_{\mathrm{TF}}$ strokes during a PARK. This global advance can be large — comparable to one inter-tile gap (≈ 59 cells; Chapter 7, Section 7.8.2) for a worst-case PARK — so the compiler must enforce a collision-safety precondition. For every ordered pair of on-path slugs (σ_i, σ_j) with σ_i upstream of σ_j at PARK start, write Δ_{ij} for the cap-to-cap separation in cells. The precondition is

$$\Delta_{ij} \;\geq\; (n_{\mathrm{stop}} + n_{\mathrm{TF}}) + n_j + \delta_{\mathrm{safe}}, \tag{9.19}$$

where n_j is the cell length of σ_j. This is a Boolean combination of integer linear inequalities over the slug positions in Σ, statically discharged at compile time because all positions are lithographically defined at every PARK checkpoint (Section 9.4.4).

9.4.4 Composition Discipline

Re-registration is the architectural mechanism for bounding cumulative position drift. The compiler enforces:

> *Every compound microarchitectural operation begins and ends at a PARK checkpoint — a state in which all operand slug positions are lithographically defined.*

Each compound starts from a known physical state (slug cap positions equal to valve positions, drift zero) and returns to a known physical state. Between PARK checkpoints, positions are approximately known (bounded by Equation 9.13); no positioning-critical action may be initiated without an intervening PARK. This is the machinery that implements the abstract-physical correspondence formalized in Chapter 8: the abstract state $S = (\Sigma, C, \nu, F)$ agrees with the physical chip configuration at every compound boundary.

9.5 Compound Microarchitectural Operations

The compiler treats five named compound operations as building blocks when lowering ISA operations, each composed of primitives, mode transitions, and PARK checkpoints. Each compound has a fixed lowering and deterministic preconditions. Table 9.10 summarizes; the rest of this section gives each in more detail.

Table 9.10: Compound microarchitectural operations.

Compound	Effect	Lowering sketch
LOAD(s, i)	Move bus slug s into register tile T_i	PARK to $J_U(i)$; SETR(i, XC)
UNLOAD(s, i)	Move slug from register T_i onto bus	SETR(i, RP); CIRC; SETR(i, BP)
INJECT(R, n)	Introduce n-cell slug from reagent stub	SETIO(IN); $n\times$ CIRC; SETIO(ID)
EJECT(s)	Remove slug s to collection	PARK s to outlet stub; SETIO(EJ); $\lvert s\rvert\times$ CIRC$(-)$; SETIO(ID)
MODULE_ACCESS(s, i)	Round-trip s to module on tile T_i	LOAD; SETR(i, MD); WAIT(t_M); SETR(i, BP); UNLOAD

LOAD. PARK the bus slug against the upstream junction $J_U(i)$ of the target tile, then SETR(i, XC): the slug is now inside the serpentine register at its downstream end. The mode-switch shift $\Delta = V_{\text{reg}} + 1$ (Chapter 7, Equation 7.11) accounts for the active-path lengthening and keeps every other on-path slug at the semantically correct position.

UNLOAD. The reverse direction: SETR(i, RP) opens the inlet V_1 alone, and a CIRC step pulls the parked slug back onto the bus through V_1; the tile then returns to BP via SETR.

INJECT, EJECT. The two I/O zone operations. INJECT enters mode IN (waste exit and reagent inlet open; Section 9.1.3, Table 9.3), runs n forward strokes, and returns to idle. EJECT parks the target slug at the outlet stub, enters mode EJ, runs $|s|$ reverse strokes to push slug material into the collection vial, and returns to idle.

MODULE_ACCESS. A LOAD into the bound tile, a transition to MD (which closes the bus-isolation valves V_5, V_6 and opens the module-side valves V_7, V_8), a WAIT for the module to complete its processing contract, a return to BP, and an UNLOAD. While the tile is in MD the bus loop is broken at that tile, so bus operations are paused for the duration of the wait. The asynchronous variant (process_async/await, Chapter 10) is enabled by module contracts that hold the slug off-chip during the long processing phase: the tile returns to BP between the short on-chip transfer phases that flank the wait, and bus operations resume on other tiles in the interim.

Every compound listed above begins and ends with all valves in a static configuration and at a PARK checkpoint, satisfying the composition discipline of Section 9.4.4.

9.6 Scheduling

The microarchitectural scheduler accepts a sequence of compound operations from the compiler and emits the timed primitive stream that drives the pneumatic controller.

9.6.1 Scheduler Obligations

1. **Preserve principle T.** Every SETR, SETMX, and SETIO is emitted only with the pump idle. Mode transitions through transient configurations AC or RP are bracketed by pump-idle windows.
2. **Respect the compliance bound.** At any program point, the count of tiles in XC mode satisfies $n_{\text{XC}}(C) \leq 2$ (Chapter 7, Equation 7.12); a planned XC transition that would violate the bound is deferred until an existing XC tile returns to BP.

3. **Insert PARK at every safety-critical boundary.** Every compound operation begins and ends with a PARK checkpoint; mode switches that would close a valve in the path of an on-path slug are preceded by PARK.
4. **Minimize wall time.** Among legal orderings, the scheduler prefers (a) the closest valid PARK anchor valve, reducing the per-PARK distance; (b) Hamming-distance minimization between successive valve configurations, reducing valve-cascade time; and (c) pipelining valve reconfiguration with CIRC transit where principle T permits.

9.6.2 Atomic Step Durations

Every compound instruction decomposes into a sequence of primitive calls, each with a deterministic duration (Table 9.11). The valve set time t_v is six times the nominal stroke period t_s and is the architectural scheduling bottleneck for any operation that toggles many valves between transit phases.

Table 9.11: Atomic step durations.

Step	Symbol	Duration	Source
SET* per valve toggle	t_v	120 ms	Demultiplexer latch (Chapter 7)
CIRC stroke at v_{nom}	t_s	20 ms	50 Hz pump cycle (Chapter 7)
CIRC stroke at v_{prec}	t_p	200 ms	5 Hz pump cycle (Chapter 7)
WAIT (coalescence dwell)	t_c	$\approx$ 10,000 ms	Coalescence timescale (Chapter 6)

9.6.3 Representative Costs

Table 9.12 gives wall-time estimates for the eight ISA-level operations of Chapter 10 as compositions of primitives and PARK checkpoints. The constants reflect typical PARK target distances (one inter-tile gap) and the mode-switch overhead at each compound boundary.

Table 9.12: Representative wall-time costs for ISA operations.

ISA operation	Primitive count (sketch)	Wall time
create(R, n)	n CIRC + 2 SETIO ($\sim$3 valves each)	$20n + 760$ ms
destroy(s)	PARK + $\lvert s\rvert$ CIRC + 2 SETIO	$20\lvert s\rvert + 620$ ms
split(s, k)	PARK + SETR + transport CIRC	$\sim$ 3 s
merge(s_1, s_2)	2$\times$ PARK + AC dwell + coalescence WAIT	$\sim$ 13 s (t_c-dominated)
mix(s)	transit CIRC + mixer cycles	$\sim$ 2 s
process(s, T_i, p)	2 LOAD/UNLOAD round trips + t_M	t_M + several s
hold(s, t)	PARK + WAIT(t)	$t + \sim$ 1 s
flush	2 I/O-zone transits	$\sim$ 2 s

PARK overhead is additive at every compound boundary, so long-distance operations are dominated by PARK time, not by valve switching. PARK costs at representative distances (Table 9.9) range from $\sim$ 1.7 s for $d = 30$ cells to $\sim$ 11 s for half-loop distances.

9.6.4 Pipelining

The Grover demultiplexer is idle during CIRC, so the scheduler pre-configures valves for the next operation during transit. The number of valves configurable during n strokes at v_{nom} is

$$V_{\text{pipe}}(n) \;=\; \left\lfloor \frac{n \cdot t_s}{t_v} \right\rfloor \;=\; \lfloor n/6 \rfloor. \tag{9.20}$$

A standard mode switch (3 valves, 360 ms) completes during any transit of $n \geq 18$ strokes. Inter-tile transits ($n \approx 59$ strokes, ≈ 1.18 s) accommodate up to nine valve toggles, enough for the BP $\rightarrow$ MD reconfiguration of one tile (7 valves) plus a partial setup of a second. Across the chip, batch reconfiguration costs $H \cdot t_v$ where H is the total Hamming distance over all tiles changing mode.

9.6.5 Bottleneck Analysis

For any ISA operation, the dominant cost depends on which of several architectural bottlenecks limits throughput (Table 9.13). The crossover between valve-bound and transit-bound regimes is at

$$n \cdot t_s \;\approx\; H \cdot t_v \;\Longrightarrow\; n \approx 6H, \tag{9.21}$$

with H the per-step Hamming distance and n the per-step transit length. Below the crossover, valve switching dominates; above it, fluid transit dominates. The scheduler's primary optimization levers are: minimize H between consecutive configurations; pipeline $H \cdot t_v$ inside the next transit; and pre-position slugs near target tiles to keep individual transit lengths small.

Table 9.13: Microarchitectural bottlenecks.

Bottleneck	Cost	Dominates when
Valve reconfiguration	$H \cdot t_v$	Frequent mode switches with short transits
Fluid transit	$n \cdot t_s$	Long-distance transport (half- or full-loop)
Bulk slug transfer	$k \cdot t_s$	Large LOAD/UNLOAD (k up to V_{reg})
Coalescence dwell	t_c	Any merge (single dwell dominates)
Module operation	t_M	PCR thermal cycles, fluorescence reads, etc.

A typical 30–50-operation assay runs on the order of minutes, with most time in coalescence and module WAITs; PARK and valve reconfiguration contribute seconds per operation. Asynchronous module invocation (process_async/await, Chapter 10) recovers parallelism otherwise lost to the serial bus.

9.7 Verification Algorithm

The dynamics of Sections 9.1.1–9.2.8, together with the compound lowerings of Section 9.5, provide a per-instruction transition relation $\delta(I)(S) = S'$ on the abstract state $S = (\Sigma, C, \nu, F)$ of Chapter 8. The verifier walks the lowered program forward over this relation, checking each precondition before δ is applied and each invariant after.

Algorithm. Let $L(\pi) = I_1, \ldots, I_m$ be the lowered program (a sequence of microarchitectural primitives interleaved with PARK macros and BRANCH nodes). Initialize $S \leftarrow S_0$ (Chapter 8, Equation 8.6). For each I_j:

1. Check $\text{Pre}(I_j)(S)$: instruction precondition (valve safety (9.8) on SETR; PARK collision safety (9.19) on PARK; compliance bound $n_{\text{XC}}(C) \leq 2$ on tile-mode changes; process_async reservation on MODULE_ACCESS).
2. Apply $S \leftarrow \delta(I_j)(S)$: position update, fragment splits with compiler-supplied fresh identifiers (Sections 9.2.2, 9.2.3), Reg $\leftrightarrow$ Bus relocations, and the post-update coalescence pass (Section 9.2.6).
3. Check $\text{Inv}(S)$: all eight invariants (Chapter 8, Section 8.4).

At a BRANCH node (Chapter 10, Section 10.4) with arms $\phi_1 \to L(\pi_1), \ldots, \phi_q \to L(\pi_q), \text{default} \to L(\pi_0)$, the verifier recurses on every arm whose predicate ϕ_j is satisfiable under the incoming ν, threading S into each. Coverage of $\prod_r \mathcal{R}_r$ by the predicates is checked once at the branch node. At program exit, $\text{dom}(F) = \emptyset$ is checked.

If any check fails on any arm, the verifier rejects with a diagnostic (offending instruction, predicate, branch path); otherwise it accepts. Each step is a decidable QF-LIA query (Chapter 8, Section 8.6); the total cost is bounded by Equation 8.15.

9.8 Soundness

An *execution trace* of $L(\pi)$ is a finite sequence of states $S_0, S_1, \ldots, S_T$ produced by applying δ from the initial state S_0, with non-deterministic readouts in ν resolved at every MODULE_ACCESS site by the actual module response.

Theorem 9.1 (Soundness of verification)**.** *If the verifier of Section 9.7 accepts $L(\pi)$, then for every execution trace of $L(\pi)$ under any concrete module-readout sequence consistent with the declared codomains $\mathcal{R}_M$, the eight invariants of Chapter 8 hold at every S_j, every instruction precondition is satisfied at every step, no future is left unawaited at program exit, and the trace terminates in bounded time.*

Proof sketch. The argument reduces to three observations.

Per-arm coverage. The verifier explores every branch arm whose predicate is satisfiable; the coverage check at each BRANCH ensures every concrete readout assignment selects exactly one arm. The static set of explored execution paths therefore contains the runtime set: any runtime trace is the verifier's symbolic trace for some path.

Per-step soundness. Each verification step checks the precondition before applying δ and the invariants after. The dynamics rules of Sections 9.1.1–9.2.8 are deterministic modulo non-deterministic readouts, which are treated as inputs. The symbolic state after step j therefore equals the runtime state for any consistent readout sequence, and any precondition or invariant the verifier checks symbolically holds at runtime.

Termination. $L(\pi)$ has finite length on every path; each microarchitectural primitive terminates in bounded physical time (Section 9.6.2); branch depth is bounded by program structure; PARK macros terminate in $n_{\text{stop}} + n_{\text{TF}}$ strokes by construction (Section 9.4). The compiler's source-level SSA pass (Chapter 10, Section 10.2) ensures no future is left unawaited at program exit. □

Compiler correctness. Theorem 9.1 discharges the operationally important consequence of verifier acceptance — that no invariant or precondition is violated — but does not formalize the lowering function $\pi \mapsto L(\pi)$ itself. A bug in the lowering that produced a low-level program with the wrong slug-routing semantics would manifest as a verification failure (e.g. a slug ending up at the wrong tile would violate a downstream MODULE_ACCESS precondition) rather than as a silent semantic error. A full lowering-correctness proof is left to a mechanized treatment.

Chapter 10

Slug-Functional Instruction Set

The user-visible abstraction of HyDRA is an instruction set of ten slug operations and two control-flow constructs, under a single-static-assignment discipline. This chapter specifies each instruction, the SSA discipline, the asynchronous module contract, and the compiler contract that bridges the ISA and the microarchitecture.

10.1 Overview

A program is a finite, conditionally-branching sequence of slug operations over named values. Slugs and futures are first-class values with stable identifiers; the compiler resolves each identifier to a physical location (bus position or register cell) at compile time. The user does not address registers, valves, or bus positions — these are compiler-managed resources.

Table 10.1 summarizes the instruction set. process and the process_async/await pair together form a synchronous/asynchronous pair for module invocation; the synchronous form is semantically equivalent to an asynchronous start followed by an immediate await, presented as a separate instruction for ergonomic convenience.

Table 10.1: The HyDRA instruction set: ten slug operations and two control-flow constructs.

Operation	Signature	Effect
create(R, n)	$R, n \to s$	Introduce volume n of reagent R
destroy(s)	$s \to$	Eject s to collection
split(s, k)	$s, k \to (s_a, s_b)$	Partition s at volume k
merge(s_1, s_2)	$s_1, s_2 \to s_3$	Coalesce s_1, s_2
mix(s)	$s \to s'$	Homogenize s
process$(\vec{s}, M, p[, r])$	$\vec{s}, M, p \to \vec{s}'$	Apply module M synchronously
process_async$(\vec{s}, M, p[, r])$	$\vec{s}, M, p \to f$	Start M, return future f
await(f)	$f \to \vec{s}'$	Retrieve outputs of future f
hold(s, t)	$s, t \to s'$	Park s for duration t
flush	$\to$	Re-establish clean carrier arc
branch$(\{\phi_j \to \pi_j\})$	$\nu \to$ one π_j	Conditional dispatch on readouts
repeat(n, π)	$n, \pi \to \pi; \pi; \cdots$	Compile-time unroll of π by n

Microarchitectural primitives (Circ, SetR, SetMx, SetIO, Wait), Park composition, and the compound microarchitectural operations LOAD, UNLOAD, INJECT, EJECT, and MODULE_ACCESS (Chapter 9, Section 9.5) do not appear at the ISA level. The user reasons in slugs, futures, and modules; the compiler reasons in cells, registers, strokes, and tile modes.

10.2 SSA Discipline

Every slug identifier and every future identifier is *defined exactly once* by the operation that produces it and remains *live* until *consumed*. After consumption (by destroy, by an operation that takes the value as input, or by await for futures), the identifier is not live and cannot be referenced again. The compiler verifies SSA conformance statically; multiple definitions, use-after-consume, dangling references, and unawaited futures (futures live at program exit) are rejected at compile time.

The user-visible state at every program point is captured by the abstract state $S = (\Sigma, C, \nu, F)$ where:

- $\Sigma : \mathrm{Id} \rightharpoonup [1, V_{\mathrm{reg}}] \times \mathit{Loc}$ is a partial map from slug identifiers to (volume, location) pairs.
- $C \in \{\mathsf{B}, \mathsf{X}\}^N$ is the tile-mode vector.
- $\nu : \mathbb{N} \to (\bigcup_M \bigcup_p \mathcal{R}_M(p)) \cup \{\bot\}$ is the readout environment; readouts are immutable once bound.
- $F : \mathrm{FutureId} \rightharpoonup (M, p, i_M)$ is the pending-future map.

SSA discipline gives invariant **I** (Chapter 8, Section 8.4): both $\mathrm{dom}(\Sigma)$ and $\mathrm{dom}(F)$ at any point in execution are exactly the set of currently-live identifiers of their respective kinds. Chapter 8, Section 8.3.3, formalizes this in terms of an Allocated/Consumed/Live partitioning of the identifier universe.

10.3 Slug Operations

Each slug operation is specified by its signature, precondition, and state transition δ; formal invariant preservation is in Chapter 8. The compiler lowers each into a sequence of microarchitectural primitives (Chapter 9).

10.3.1 create

Inject a volume-n slug of reagent R. The reagent identity R maps to a physical reagent stub (dedicated or multiplexer-selected; Chapter 7).

Pre. $n \in [1, V_{\mathrm{reg}}]$; the I/O zone is clear of slugs at the injection cells. δ**.** Fresh s; $\Sigma \leftarrow \Sigma \uplus \{s \mapsto (n, \mathrm{Bus}(x_{\mathrm{new}}))\}$; all other live slugs shift $+n$ as the n forward strokes of INJECT (Chapter 9, Section 9.5) push reagent in via the reagent stub and displace equal carrier oil to waste.

10.3.2 destroy

Eject s to the collection vial.

Pre. $s \in \mathrm{dom}(\Sigma)$. δ**.** Let n_s be the volume of s (i.e. $\Sigma(s) = (n_s, \ell_s)$). s is consumed; all other live slugs shift $-n_s$ as the n_s reverse strokes of EJECT (Chapter 9, Section 9.5) draw fresh carrier in via the carrier-in stub and push slug material out to collection.

10.3.3 split

Partition s of volume n into s_a (k cells) and s_b ($n - k$ cells).

Pre. $s \in \mathrm{dom}(\Sigma)$; $k \in [1, n-2]$. The upper bound excludes the degenerate $k = n - 1$ case, whose one-cell trailing fragment s_b has size equal to the architectural cell length, leaving no carrier-oil margin to discriminate s_b from the adjacent cells under the inter-slug-separation invariant D. δ**.** Fresh s_a, s_b; s consumed. Outputs are adjacent on the bus after the split valve closes; subsequent circulation separates them.

10.3.4 merge

Coalesce s_1 and s_2.

Pre. $s_1, s_2 \in \mathrm{dom}(\Sigma)$; $s_1 \neq s_2$; $n_1 + n_2 \leq V_{\mathrm{reg}}$. δ. Fresh s_3 with $n_3 = n_1 + n_2$; s_1, s_2 consumed. Lowering: two PARKs bringing each input against opposite faces of a shared valve, a valve opening, and a coalescence wait $t_c = 10$ s (Chapter 6).

10.3.5 mix

Homogenize the internal contents of s (typically after a merge combining chemically-different inputs).

Pre. $s \in \mathrm{dom}(\Sigma)$. δ. Fresh s' with same volume; s consumed. Lowering: switch mixer contour to active, transport through the chaotic-advection serpentine, restore to isolated.

10.3.6 process, process_async, await

These three operations invoke external modules. A module M is declared at chip bring-up with contract $(\vec{a}_M, \vec{n}'_M, \mathcal{P}_M, \mathcal{R}_M)$: input arity $\vec{a}_M$; output-volume function $\vec{n}'_M(\vec{n}, p)$ satisfying $\sum_j n'_j \leq \sum_i n_i$ (volume-preserving or volume-reducing); parameter domain $\mathcal{P}_M$; finite readout codomain $\mathcal{R}_M$. Each tile T_i is bound to exactly one module at bring-up; the binding is fixed for the chip's lifetime. A representative module declaration in the source language:

```
module THERMOCYCLER on tile T2 {
  inputs   : 1 slug, volume in [1, V_reg]
  params   : (temp: int [25..98], minutes: int [1..120])
  outputs  : 1 slug, same volume
  readout  : none
  duration : minutes * 60 + 30 (seconds)
}
```

The compiler ingests one such declaration per tile; the assignment of physical tile to module is a chip-bring-up configuration item outside the user's program.

process$(\vec{s}, M, p[, r]) \to \vec{s}'$ — **synchronous invocation.**

Pre. Each $s_i \in \mathrm{dom}(\Sigma)$, distinct; $|\vec{s}| = a_M$; $p \in \mathcal{P}_M$. δ. Transfer $\vec{s}$ off-chip; block bus until M returns; retrieve $\vec{s}'$; consume $\vec{s}$; if r given, bind $\nu(r) \in \mathcal{R}_M(p)$.

process_async$(\vec{s}, M, p[, r]) \to f$ — **asynchronous invocation.**

Pre. As process; additionally tile i_M in B mode with no pending future. δ. Transfer $\vec{s}$ off-chip; reserve tile i_M; produce fresh f with $F(f) = (M, p, i_M)$; consume $\vec{s}$; bus released for other operations.

await$(f) \to \vec{s}'$ — **synchronize on future.**

Pre. $f \in \mathrm{dom}(F)$. δ. Block until M reports completion; retrieve $\vec{s}'$; release tile reservation; if readout declared, bind $\nu(r) \in \mathcal{R}_M(p)$; consume f.

Synchronous process is semantically process_async; await; the asynchronous form merely exposes the pipeline.

Asynchrony and Pipeline Parallelism

Module contracts partition into a short transfer phase (seconds, bus locked) and a long pending phase (minutes to hours, slug held off-chip). Between process_async and await, arbitrary bus operations may proceed; up to $N = 10$ futures can be pending concurrently.

For a streaming workload with per-batch bus preparation T_{prep}, module duration T_{module}, and bus post-processing T_{measure}, the synchronous schedule processes one batch every

$$T_{\mathrm{serial}} = T_{\mathrm{prep}} + T_{\mathrm{module}} + T_{\mathrm{measure}}.$$

The asynchronous schedule overlaps the bus phases of one batch with the module phase of those preceding it. With N tile-bound module instances available for parallel execution, the steady-state per-batch time is

$$T_{\text{async}} = \max(T_{\text{prep}} + T_{\text{measure}},\ \ T_{\text{module}}/N)\,, \tag{10.1}$$

bus-bound when $T_{\text{prep}}+T_{\text{measure}} > T_{\text{module}}/N$ and module-bound otherwise. The asymptotic throughput speedup is

$$\frac{T_{\text{serial}}}{T_{\text{async}}} \ \to\ 1 + \frac{T_{\text{module}}}{T_{\text{prep}} + T_{\text{measure}}} \quad \text{as } N \to \infty,$$

bounded above by $N \cdot T_{\text{serial}}/T_{\text{module}}$ in any finite-tile architecture. For PCR or thermocycler workloads with $T_{\text{module}} = 30$ min and $T_{\text{prep}} + T_{\text{measure}} = 1$ min, the asymptotic limit is $31\times$. With $N = 10$ tiles, the schedule is module-bound (since $T_{\text{module}}/N = 3$ min > 1 min), giving an achievable speedup of $T_{\text{serial}}/(T_{\text{module}}/N) = 31/3 \approx 10\times$.

10.3.7 hold

Park s for duration t (binding kinetics, incubation, synchronization).

Pre. $s \in \text{dom}(\Sigma)$; $t \in \mathbb{N}_{>0}$ (ms). **δ.** Fresh s' of same volume; s consumed. Lowering: PARK s into a register, WAIT(t), UNLOAD back to bus.

10.3.8 flush

Re-establish a clean carrier arc between protocol stages.

Pre. Flush region clear of slugs. **δ.** Σ, ν, F unchanged. Lowering: open fresh-carrier and waste-out stubs, circulate to displace the flush volume, close both.

10.4 Control Flow

Two control-flow constructs extend the straight-line ISA.

10.4.1 branch

Execute one of several sub-programs based on the readout environment ν.

Syntax.

$$\textsf{branch}\ \{\ \phi_1 \to \pi_1,\ \ldots,\ \phi_q \to \pi_q,\ \text{default} \to \pi_0\ \}$$

Each ϕ_j is a quantifier-free predicate in linear integer arithmetic (QF-LIA) over bound readouts; each π_j is a sub-program.

Pre. Every readout r appearing in any ϕ_j is bound ($\nu(r) \neq \perp$). The arms together with default cover $\prod_r \mathcal{R}_r$.

δ. At runtime, the unique arm π_j whose $\phi_j(\nu)$ holds (or π_0 if none do) executes; the others are abandoned.

SSA Coherence. Each arm executes in its own SSA scope; identifiers defined inside arm π_j are not visible outside it. Any identifier defined *before* the branch that must outlive the branch must be assigned the same name in every arm (*coherent out-identifier discipline*), so that the identifier has a well-defined

value on the join. For futures specifically, every arm must either await each pre-branch future or leave it untouched — mixed treatment is a compile error.

Verification. The compiler verifies every arm independently, threading the pre-branch state into each. Coverage is checked once at the branch node. A program with b branch nodes and maximum arity q has at most $(q+1)^b$ execution paths, all exhaustively verified.

10.4.2 repeat

Unroll a sub-program a compile-time-constant number of times, binding fresh identifiers on each iteration.

Syntax.

$$\mathsf{repeat}(n, \pi) \to \pi_1; \pi_2; \ldots; \pi_n.$$

Pre. n is a literal or parameter bound at ISA-issuance time (not by a runtime readout). The body π is independently valid.

δ. The compiler unrolls $\mathsf{repeat}(n, \pi)$ to n sequentially-concatenated copies of π, with fresh identifiers in each copy. After unrolling, the expanded program is verified as a straight-line sequence.

Rationale. Bounding iteration at compile time preserves the finite-DAG program structure required by the tractability argument (Chapter 8). Data-dependent loops — where iteration count depends on a runtime readout — would render program length unbounded and break the polynomial-resource verification bound. They are deliberately excluded.

Interaction with branch. repeat may contain branch inside its body; since unrolling predates branch analysis, the resulting expanded program is a conditional DAG with branch depth $b' \leq b_\pi \cdot n$ where b_π is the branch depth of π. Branch depth grows linearly in the unroll count, not exponentially, because unrolled copies are sequenced rather than nested.

10.5 Compiler Contract

The compiler bridges the slug-functional ISA and the microarchitecture. Its obligations are:

1. **SSA verification.** Every slug identifier and every future identifier is defined exactly once and used only while live. Multiple definitions, use-after-consume, dangling references, unawaited futures (futures live at program exit), or values outliving a branch without coherent handling in every arm are statically rejected.
2. **Register allocation.** Each live slug is assigned a physical location (bus position or register cell). The compiler tracks register occupancy, reuses registers across non-overlapping slug lifetimes, respects reserved tiles (tiles with pending futures, invariant **A**), and rejects programs whose maximum live-slug count exceeds N registers plus bus capacity.
3. **Module placement and multi-input realization.** Modules are bound to physical tiles at bring-up. Each tile has one module-side fluid interface; multi-input $\mathsf{process}([s_1, \ldots, s_a], M, p)$ is realized by the compiler as a single payload of total volume $\sum_i n_i \leq V_{\text{reg}}$, built by serially loading each input into the tile's register (using MERGE semantics to combine in place) and then transitioning to MD mode for transfer. Multi-output modules retrieve outputs serially via separate MD round trips in the module's declared output order.
4. **Lowering.** Each ISA operation lowers to a sequence of microarchitectural primitives (Circ, SetR, SetMx, SetIO, Wait) that effects the same abstract-state transition. The lowering inserts Park composite sequences to position slugs with lithographic precision before every critical valve event, respecting all microarchitectural preconditions (valve-safety equation (9.8), PARK collision-safety (9.19), $n_{\text{XC}} \leq 2$).

5. **Verification of lowered program.** After lowering, the resulting low-level program is verified by forward symbolic execution (Chapter 8, Section 8.6). The verifier checks all preconditions and invariants (**V, T, P, D, B, C, I, A**) at every program point on every branch arm.

The compiler may reject programs that exceed physical resource limits (single-slug volume $> V_{\text{reg}}$; simultaneous live slugs exceeding N plus bus capacity; simultaneous pending futures exceeding N; combined multi-input payload exceeding V_{reg}) or violate SSA discipline.

10.6 Example: Serial Dilution

Figure 10.1 shows a three-stage serial dilution at the 3/7 ratio — a standard biochemical workflow that exercises non-dyadic split ratios. The program produces s_3 of concentration $(3/7)^3 \approx 0.079$ relative to the initial stock.

```
slug stock      = CREATE(SAMPLE, 9)
slug buf        = CREATE(BUFFER, 12)
slug a1, rest1  = SPLIT(buf, 4)        // a1 = 4 cells, rest1 = 8 cells
slug a2, a3     = SPLIT(rest1, 4)      // a2 = 4, a3 = 4
slug t1, r1     = SPLIT(stock, 3)      // t1 = 3 (transfer), r1 = 6 (residue)
slug p1         = MERGE(t1, a1)
slug s1         = MIX(p1)
slug t2, r2     = SPLIT(s1, 3)
slug p2         = MERGE(t2, a2)
slug s2         = MIX(p2)
slug t3, r3     = SPLIT(s2, 3)
slug p3         = MERGE(t3, a3)
slug s3         = MIX(p3)
DESTROY(r1); DESTROY(r2); DESTROY(r3)
```

Figure 10.1: Three-stage serial dilution at the 3/7 ratio in the HyDRA ISA. Each stage takes 3 cells of the previous slug and merges with a 4-cell buffer aliquot to yield 7 cells at 3/7 the prior concentration; final strength $(3/7)^3 \approx 0.079$. The example exercises non-power-of-two split ratios, confirming that the integer-stroke abstraction imposes no restriction to dyadic fractions — the only constraint is $k \in [1, n-2]$ on each split. Maximum live-identifier count is 5.

Every intermediate has a name and a statically-analysable lifetime; the register allocator can observe that at most five slugs are live at any point, well under the ten-tile register budget. Diluent preparation, aliquoting, and mixing are all standard operations; no valve- or register-level primitives appear in the source. A conditional variant is obtained by replacing the final destroy operations with a branch on a concentration readout; the SSA structure is preserved, and each arm is independently verified.

Chapter 11

Completeness and Assay Coverage

The formal model of Chapter 8 establishes *soundness*: every program that passes compile-time verification executes without safety violations. This chapter establishes *coverage*: the target class of biochemical assays identified in Chapter 4 is expressible in the HyDRA ISA within the architecture's finite resource envelope. Coverage is demonstrated by translating representative assays into ISA programs, tracing their live-identifier counts and volume requirements, and confirming that all resource bounds are respected.

11.1 Target Assay Class

The target class comprises five representative biochemical workflows:

- **PCR.** Polymerase chain reaction: template and master mix combined, then cycled thermally for 30–40 iterations.
- **Immunoassay.** Antigen and antibody combined, incubated for binding kinetics, then fluorometrically measured.
- **Serial dilution.** Stock reagent successively diluted by merging with buffer aliquots.
- **Protein crystallization.** Protein and precipitant combined in multiple ratio/temperature conditions, incubated, and held for nucleation windows.
- **Glucose assay.** Clinical-scale glucose measurement with enzymatic chemistry and fluorescence readout; used as the *worked example* in Section 11.5.

All five are expressible as finite conditional DAGs over the primitive operations in Chapter 10. Table 11.1 summarizes each, its maximum live-identifier count, and the ISA operations it requires.

Table 11.1: Coverage of the target assay class. d is the number of dilution stages; c is the number of crystallization conditions. All max live-identifier counts $\leq N = 10$; all single-slug volumes $\leq V_{\text{reg}}\gamma \approx 250$ nL.

Assay	Max live ids	ISA ops	Characteristic pattern
PCR	2	~ 12	create $\times 2$, merge, process cycling
Immunoassay	2	~ 18	create $\times 2$, merge, process incubate, fluoro process
Serial dilution	$\leq d + 2$	$\sim 4d + 2$	create $\times 2$, repeated split + merge
Protein crystallization	$\leq c$	$\sim 6c$	create $\times (c + 1)$, aliquoted split/merge
Glucose assay	3	12	Worked example, Section 11.5

11.2 Sketch Programs

The live-id and operation counts in Table 11.1 are not just asserted; each row corresponds to a verified ISA program whose dependency DAG fits in the architecture's register file. The serial-dilution case at

$d = 3$ is fully worked in Chapter 10, Figure 10.1, with peak $|\Sigma| = 5$ tracing the $d + 2$ pattern. The remaining three are sketched here in compact form; the glucose assay receives the full live-id trace in Section 11.5.

PCR. Template DNA and master mix are merged into a single thermocycler payload, then cycled in place by the module:

```
slug template = CREATE(SAMPLE, 5)
slug mix      = CREATE(MASTER_MIX, 10)              // peak |Sigma| = 2
slug payload  = MERGE(template, mix)
slug cycled   = PROCESS(payload, THERMOCYCLER, (cycles=35))
DESTROY(cycled)
```

The PCR cycling itself is performed inside the THERMOCYCLER module, not by the ISA: a single process call with a cycle-count parameter; the live-id count is bounded at the moment both inputs are created and unmerged.

Immunoassay. Antigen and antibody are merged, incubated for binding kinetics, then read fluorometrically:

```
slug antigen   = CREATE(SAMPLE, 5)
slug antibody  = CREATE(REAGENT, 5)                 // peak |Sigma| = 2
slug bound     = MERGE(antigen, antibody)
slug incubated = HOLD(bound, 1800000)               // 30 minutes
slug measured  = PROCESS(incubated, FLUOROMETER, (520, 600, 100), r_signal)
DESTROY(measured)
```

The peak live count is 2, attained between the two creates. The hold replaces a process call when no module action is needed during incubation.

Protein crystallization. A protein stock is split into c aliquots and each is combined with a different precipitant condition:

```
slug protein = CREATE(SAMPLE, 4*c)
REPEAT(c) {
  slug aliq, rest = SPLIT(protein, 4)
  slug precip     = CREATE(PRECIPITANT_i, 4)
  slug cond_i     = MERGE(aliq, precip)
  slug held_i     = HOLD(cond_i, t_nuc)             // each iteration: |Sigma| <= c+1
  DESTROY(held_i)
  // protein is rebound to rest for next iteration
}
DESTROY(protein)
```

After unrolling, the live-id count at iteration j is at most $j + 2$ if hold-then-destroy operations within each iteration are deferred to the end (worst-case scheduling), or 2 if each iteration's destroy is sequenced before the next split (the default schedule the compiler emits). Under the default schedule the peak is bounded by a constant independent of c, but for the table entry we conservatively report the worst-case $\leq c$ for $c \leq N = 10$.

11.3 Register Sufficiency

Two bounds together establish register sufficiency for the target class.

Register count vs. live-slug count. The maximum number of simultaneously live slugs equals the DAG width (maximum antichain over the data-dependency DAG). For the target assays, the DAG width

is at most $N = 10$, with serial dilution at $d = 8$ stages being the tightest case (live count $d + 2 = 10$). Ten registers accommodate this exactly; the bus can hold additional slugs in transit, but the compiler does not rely on bus-resident storage for liveness.

Register capacity vs. slug volume. The maximum single-reagent volume in the target class is ~ 400 nL (serial dilution initial stock). Since merge requires $n_1 + n_2 \leq V_{\text{reg}}$ and create requires $n \leq V_{\text{reg}}$, any single slug is bounded by $V_{\text{reg}}\gamma \approx 250$ nL. For assays requiring larger volumes, the compiler can assemble them through sequential merge operations across multiple preparation batches, though this introduces per-batch variance that assay-level calibration must accommodate.

11.4 Data Dependencies and Sequencing

Each edge in the assay's data-dependency DAG corresponds to passing a slug identifier from a producer operation to a consumer operation — exactly what SSA naming makes explicit. The topological ordering of the source program guarantees that every input identifier is live (defined and not yet consumed) at every use site.

Multi-use intermediates are handled by split: if a stored slug s must be split for use in multiple downstream operations, split(s, k) produces two fresh identifiers s_a, s_b of disjoint volume; both flow forward in the DAG. Repeated splitting realizes arbitrary multi-way fan-out, with volume accounting enforced by invariant **C** at every step.

The compiler tracks register allocation, slug positions, and live-identifier sets as part of its lowering pass. At each program point, the abstract state $S = (\Sigma, C, \nu, F)$ is fully determined, and all preconditions are checkable by direct computation over integers (Chapter 8, Section 8.6).

11.5 Worked Example: Glucose Assay

The clinical glucose assay combines a patient sample with a buffer and an enzyme substrate, incubates the mixture at 37°C for 30 minutes, then measures fluorescence at the appropriate excitation and emission wavelengths. A conditional reflex variant re-runs the assay if the initial measurement falls outside a diagnostic range.

11.5.1 Dependency DAG

The data-dependency DAG has twelve operations: four creates (sample, buffer, enzyme, substrate), three merges, one thermocycler process, one split (aliquot + waste), one fluorometer process, and two destroys. Its global width (maximum antichain) is 4, attained by the four creates, but a topological ordering that interleaves creates with their immediate merges reduces simultaneously-live counts: the ordering chosen below keeps peak live-identifier count at 3.

11.5.2 ISA Program

The compiler emits one slug-functional ISA operation per DAG vertex, choosing a topological ordering (sample-buffer subtree first, then the enzyme-substrate subtree) that interleaves creation and consumption to minimize peak live count:

11.5.3 Live-Identifier Trace

Tracing the ISA program line by line, Table 11.2 shows the live-identifier set after each operation.

Peak live-identifier count is 3, well within the $N = 10$ register budget, attained after create(substrate) when the partially-merged `diluted` sits alongside the still-fresh `enzyme` and `substrate` awaiting their own merge. Volume bounds $n \leq V_{\text{reg}}$ are trivially satisfied (all slugs ≤ 6 cells).

```
slug sample      = CREATE(SAMPLE, 2)
slug buffer      = CREATE(BUFFER, 2)
slug diluted     = MERGE(sample, buffer)                          // 4 cells
slug enzyme      = CREATE(ENZYME, 1)
slug substrate   = CREATE(SUBSTRATE, 1)
slug enz_mix     = MERGE(enzyme, substrate)                       // 2 cells
slug reaction    = MERGE(diluted, enz_mix)                        // 6 cells
slug incubated   = PROCESS(reaction, THERMOCYCLER, (37, 30))      // 6 cells
slug aliquot, waste = SPLIT(incubated, 3)                         // 3 + 3
slug measured    = PROCESS(aliquot, FLUOROMETER, (520, 600, 100), r_abs)
DESTROY(measured)
DESTROY(waste)
```

Figure 11.1: Glucose assay in the HyDRA ISA. Peak live-identifier count is 3 (after CREATE of substrate: {diluted, enzyme, substrate}).

Table 11.2: Live-identifier trace for the glucose assay program.

Operation	Live identifiers after	$\lvert\Sigma\rvert$
create(sample)	{sample}	1
create(buffer)	{sample, buffer}	2
merge(sample, buffer)	{diluted}	1
create(enzyme)	{diluted, enzyme}	2
create(substrate)	{diluted, enzyme, substrate}	3 (peak)
merge(enzyme, substrate)	{diluted, enz_mix}	2
merge(diluted, enz_mix)	{reaction}	1
process(reaction, ...)	{incubated}	1
split(incubated, 3)	{aliquot, waste}	2
process(aliquot, FLUOROMETER, ...)	{measured, waste}	2
destroy(measured)	{waste}	1
destroy(waste)	$\emptyset$	0

11.5.4 Conditional Reflex Variant

A diagnostic reflex — re-running the assay at a different substrate concentration if the initial reading is below a threshold — is expressed by replacing the final destroys with a branch:

In the low arm, `waste` is consumed by the re-run process call (which produces `rerun`), and `rerun` is then destroyed alongside `measured`; in the default arm, both `measured` and `waste` are destroyed directly. SSA coherence holds: both arms consume `measured` and `waste` exactly once, no slug needs to outlive the branch (so the coherent-out-identifier discipline imposes no constraints), and the joined post-branch state is $\Sigma = \emptyset$.

11.5.5 Async-Pipelined Batch Variant

For batch throughput, the 30-minute thermocycler step can be overlapped with the bus-side phases of *other* batches by replacing process with the asynchronous process_async/await pair:

Per-batch wall time drops from $T_{\text{prep}}+T_{\text{therm}}+T_{\text{fluoro}}+T_{\text{destroy}}$ to $\max(T_{\text{prep}}+T_{\text{fluoro}}+T_{\text{destroy}},\ T_{\text{therm}}/N)$ (equation 10.1). With $T_{\text{therm}} = 30$ min and $T_{\text{prep}} + T_{\text{fluoro}} + T_{\text{destroy}} \approx 1$ min, the asymptotic limit ($N \to \infty$) is $1 + T_{\text{therm}}/(T_{\text{prep}} + T_{\text{fluoro}} + T_{\text{destroy}}) = 31\times$, and the achievable speedup with $N = 10$ tile-bound thermocyclers is $\approx 10\times$ (module-bound, with each of 10 modules doing one cycle every 3 minutes). The asymptote is approached as N scales up to $\lceil T_{\text{therm}}/(T_{\text{prep}} + T_{\text{fluoro}} + T_{\text{destroy}})\rceil = 30$ tiles, beyond which the bus saturates.

```
BRANCH {
  nu(r_abs) < 100 -> {
    // Low reading: re-run at higher substrate concentration
    slug rerun = PROCESS(waste, FLUOROMETER, (520, 600, 200))
    DESTROY(measured)
    DESTROY(rerun)
  },
  default -> {
    DESTROY(measured)
    DESTROY(waste)
  }
}
```

Figure 11.2: Conditional reflex branch for the glucose assay. Two arms (low-reading reflex and default); `measured` and `waste` are each consumed exactly once on every path.

```
REPEAT(k) {
  slug sample = CREATE(SAMPLE, 2)
  // ... (buffer, enzyme, substrate, merges as before) -> reaction
  future f          = PROCESS_ASYNC(reaction, THERMOCYCLER, (37, 30))
  // Bus released here; the scheduler may begin the next batch's prep
  // on a different thermocycler-bound tile while f is pending.
  slug[] outs       = AWAIT(f)
  slug incubated    = outs[0]                  // single-output module
  // ... (fluorometer, destroys as before)
}
```

Figure 11.3: Async-pipelined batch variant. Each iteration launches its thermocycler step asynchronously and releases the bus until await. With multiple thermocycler-bound tiles available, the compiler can schedule the bus-side phases of iteration $i+1$ during the long f_i wait of iteration i, recovering pipeline parallelism that the synchronous form would serialize.

11.6 Non-Goals and Failure Modes

Some programs will fail compile-time verification. Table 11.3 summarizes the principal failure modes and what each signals.

Each failure mode is reported at a specific ISA operation with a concrete resource bound, giving the programmer an actionable diagnostic rather than an opaque rejection.

11.7 Summary

The target assay class $\mathcal{A}$ is expressible within HyDRA's finite resource envelope. Concretely, for every assay $A \in \mathcal{A}$, there exists an ISA program π_A such that

$$
\begin{aligned}
|\Sigma|_{\max}(\pi_A) &\le N = 10, \\
\max_{s \in \Sigma(\pi_A)} n_s &\le V_{\text{reg}} = 416 \text{ cells} \approx 250 \text{ nL}, \\
\text{verification cost}(\pi_A) &= O(m \cdot (q{+}1)^b \cdot |\Sigma|^2_{\max}) \le 6 \times 10^5 \text{ integer comparisons.}
\end{aligned}
$$

The glucose assay is worked in full detail, tracing every live-identifier count and confirming compliance with all eight invariants; the async-pipelined variant demonstrates that process_async/await delivers order-of-magnitude throughput improvement on module-bound workloads. The architecture is therefore sufficient for its stated class, answering research question RQ3 (Chapter 3).

Table 11.3: Compile-time failure modes.

Failure	Meaning
Max live count $> N$	Program requires more simultaneously-live slugs than the register file supports. Programmer must refactor to reduce DAG width or accept that the assay is out of scope.
Slug volume $> V_{\text{reg}}$	A single slug cannot be larger than the register capacity. For biochemistry needing larger volumes, the programmer assembles volume through sequential merges across multiple batches.
Unbounded iteration	repeat with a non-literal count; branch arms with runtime-dependent nesting depth. Such programs are rejected as being outside the decidable fragment.
Incoherent branch	An identifier defined before branch and referenced after the branch is assigned in some arms but not others. Programmer must add appropriate destroys or re-define the identifier in every arm.
Unawaited future	A process_async whose future is not reached by await on every execution path. Usually a programmer error; sometimes indicates an intent to discard the future's result without retrieving it, which must instead be done with an explicit await followed by destroy.

Chapter 12

Results and Discussion

This chapter consolidates what has been established, contrasts HyDRA against prior work, and discusses the engineering trade-offs committed to by the design.

12.1 Summary of Technical Results

R1 (RQ1). Integer-valued discretization is achievable: principles V, T, P impose a discrete state space on the continuous substrate, with all architectural quantities ($\ell_v = 200$ μm, $\gamma = 0.6$ nL, $V_{\text{reg}} = 416$ cells, $L_{\text{bus}} = 619$ cells, $n_{\text{XC}} \leq 2$) derivable from them (Chapters 5, 7).

R2 (RQ2). The fluid system admits a non-empty joint window: wall-film stability and on-demand coalescence hold simultaneously for $\Gamma/\Gamma_{\text{sat}} \in [4\%, 14\%]$, bounded by Alexander–de Gennes steric repulsion above and the transport-phase stability requirement below, confirmed by three independent literature sources [28, 34, 10] (Chapter 6).

R3 (RQ3). The target assay class (PCR, immunoassay, serial dilution, protein crystallization, glucose assay) is expressible as finite conditional DAGs within the resource envelope ($N = 10$ registers, $V_{\text{reg}}\gamma \approx 250$ nL per slug), with peak live-identifier counts from 3 (glucose) to 10 (serial dilution at $d = 8$) (Chapter 11).

R4 (RQ4). Safety-relevant properties are ahead-of-time decidable in polynomial resources: eight invariants, each a Boolean combination of integer linear inequalities, verified at cost $O(m \cdot (q{+}1)^b \cdot |\Sigma|^2_{\max}) < 6 \times 10^5$ integer comparisons for realistic workloads (Chapter 8).

R5 (RQ5). Asynchronous module contracts recover pipeline parallelism: `process_async`/`await`, sound under invariant **A**, delivers throughput approaching $1 + T_{\text{module}}/(T_{\text{prep}} + T_{\text{measure}})$ in the bus-bound asymptote, with the finite-N ceiling at $N \cdot T_{\text{serial}}/T_{\text{module}}$ in the module-bound regime — $\approx 10\times$ speedup at $N = 10$ for the 30-minute-thermocycle / 1-minute-bus workload, asymptotic to $31\times$ as N scales (Chapters 10, 11).

12.2 Comparison with Prior Work

Each prior system makes a defensible design trade. HyDRA's distinct position is the choice of slug flow: one-dimensional (unlike EWOD's 2D grid), natively identity-bearing (unlike continuous flow), and compatible with mature Quake-valve PDMS (unlike more specialized substrates).

Table 12.1: Detailed comparison of HyDRA with prior programmable-microfluidic abstractions.

System	Contribution	HyDRA's Difference
BioCoder [4]	Protocol-description DSL	Specifies substrate-level state, enabling compile-time safety
BioStream [40]	Dataflow for continuous-flow	Discrete integer state, not continuous concentration fields
AquaCore [3]	CF architecture with PEs	Integer-valued ISA, not continuous dataflow
BioScript [31]	Typed chemistry on EWOD	Static checks extend to substrate-level state, not just chemistry
Puddle [45]	EWOD runtime error detection	Ahead-of-time, not runtime-only guarantees
BioWare-CFP [39]	Modular CFP	ISA and formal model rather than module interoperability

12.3 Engineering Trade-Offs

The architecture commits to six design choices that each trade benefit against cost. Table 12.2 summarises them; the paragraphs that follow elaborate each.

Table 12.2: Engineering trade-offs committed to by the design.

Trade-off	HyDRA's choice	Cost / alternative
Bus count	Single bus, single pump	Serial bus operations; dual-bus variant relaxes n_{XC} at doubled pneumatic complexity
Register count	$N = 10$ tiles	Matches realistic DAG widths; larger N extends to wider antichains via parametric scaling
Slug volume	$V_{\mathrm{reg}}\gamma \approx 250$ nL	Matched to droplet-scale; bulk-scale requires sequential merges with per-batch variance
State arithmetic	Integer cells and strokes	Sub-cell rational splits unavailable; worst-case quantization error one cell (0.24%)
Identifier discipline	SSA	Mutable state would defeat tractability bound (alias analysis required)
Iteration	Bounded only (repeat, branch)	Data-dependent loops excluded; titration-to-endpoint out of scope

Single bus vs. dual. The single-pump specification serializes bus-side operations; asynchronous module contracts recover parallelism across modules but not across the bus itself. A dual-bus variant placed on diametrically-opposite pumps would halve the worst-case PARK distance, relax the compliance bound to $n_{\mathrm{XC}} \leq 4$, and admit bus-side parallelism at the cost of doubled pneumatic complexity and pump synchronization (Chapter 14, Section 14.2.1).

$N = 10$ vs. larger register file. $N = 10$ matches the DAG width of realistic assays; the tightest case is serial dilution at $d = 8$ ($10 = N$ exactly). The formal invariants are parametric in N, so a 20- or 40-tile variant is a pure geometric extension fitting within the ISO large-chip envelope.

250-nL slug vs. larger. $V_{\mathrm{reg}}\gamma \approx 250$ nL is matched to droplet-scale biochemistry. Larger single-reagent volumes are assembled through sequential merges across multiple batches, at the cost of per-batch variance the assay protocol must accommodate; μL-scale bulk biochemistry is out of scope.

Integer state vs. rational. Every volume and position is an integer. Fractional splits ("45% of s") are expressed by cell-level sizing; worst-case discretization error is one cell (0.24% of a full register) — well below typical assay variance.

SSA vs. mutable state. The discipline is the price of compile-time verification: without SSA, the finite-state tractability argument of Chapter 8 would require alias analysis and mutable-state reasoning, defeating the tractability bound.

Bounded iteration. Data-dependent loops are excluded; titrate-to-endpoint assays and similar lie outside the present architectural envelope. `branch` and `repeat` together cover conditional reflex and bounded-count variants within the decidable fragment.

12.4 Open Design Choices

Three choices are deliberately left open at the architectural level:

- **Mixing-contour internal geometry.** Hairpin count and segment lengths are fabrication-time decisions; the architectural requirement (volume-preserving homogenization) admits any compliant geometry.
- **Reagent-stub multiplexing.** Selection among multiple reagents is delegated to an external selector valve outside the architectural scope.
- **Per-tile module binding.** Tiles may be bound homogeneously (one module type, multiple instances for parallel module-bound throughput) or heterogeneously (different module types for richer assay classes); this is a chip-fabrication configuration item.

None of these affects the validity of the specification or the coverage argument.

Chapter 13

Conclusions

The thesis set out to answer whether an instruction-set-style abstraction is achievable for microfluidic biochips. HyDRA is the answer: a programmable substrate on which aqueous-slug handling is exposed as a discrete integer-valued instruction set, and on which every safety-relevant property is decidable ahead of time at cost polynomial in program size and live-slug count for bounded branch depth.

13.1 Significance

HyDRA's distinct position in the design space arises from three architectural properties, none individually unprecedented but not previously combined.

1. **Substrate-level static reasoning.** Substrate properties — valve safety, volume conservation, register capacity, slug separation — are decided at compile time, unlike EWOD type systems that address chemistry only or continuous-flow architectures that cannot reduce state to integers.
2. **Ahead-of-time, not runtime, guarantees.** Programs either pass verification (and execute without covered failures) or are rejected with actionable diagnostics. Execution-time failure modes covered by the invariants are eliminated by construction, unlike Puddle's runtime correction model.
3. **Module-level pipeline parallelism on serial hardware.** Asynchronous module contracts decouple on-chip operations from off-chip module operations; throughput approaches the module bottleneck alone for module-dominated workloads.

The architecture is also *compositional*: any sub-program valid in isolation remains valid when embedded, because all invariants hold at every ISA boundary. The specification is *layered and dependency-ordered*: principles, fluid regime, physical architecture, formal model, microarchitecture, and ISA each depend only on what precedes them, and a change at one layer propagates predictably — a fabrication-geometry change affects only architectural parameters; an ISA extension affects only the lowering. The architecture is *substrate-specific but conceptually portable*: the patterns (SSA ISA, finite-integer verification, asynchronous module contracts) are independent of slug flow and could inform programmable abstractions on other substrates.

13.2 Limitations

The thesis carries the following scope limitations.

Specification, not artifact. It is a specification, not a fabricated device: no HyDRA chip has been built, no program executed, no claim empirically measured. The natural continuations — single-tile prototype, reference compiler, machine-checked formalization, calibration protocol — are catalogued in Chapter 14.

Conditional physical feasibility. The physical-feasibility argument is conditional on one element: the per-chip Γ_{sat} calibration for the specific PFPE-PEG batch. This is a standard pendant-drop tensiometry measurement (Chapter 14), not an open physical question, but the architecture cannot be deployed without it. Every other physical claim is either cited or derived from cited values.

One unverified scaling step. The $k = 4 \to 7$ pneumatic-demultiplexer extrapolation [17, 16] is the single architectural element flagged as unverified: Grover's demonstrated devices operate at $k = 4$; HyDRA's 83-valve scale requires $k = 7$. Scaling is logarithmic in valve count, but the step would benefit from empirical validation, addressed in Chapter 14.

Substrate specificity. The operating-window analysis is specific to the HFE-7500 / Fluosurf-C / Aquapel-treated PDMS fluid system; alternative systems require a fresh regime analysis. The architectural patterns transfer; the numerical bounds do not.

Bounded computation. Data-dependent loops are excluded, so titration-to-endpoint and adaptive-feedback assays lie outside the present scope. The architecture assumes a single peristaltic pump; a dual-pump variant is future work.

Pen-and-paper proofs. The eight invariants are proved by structural induction over the ISA operations and case analysis on mode transitions; the proofs use no unstated lemma but are not mechanized in a proof assistant.

13.3 Conclusion

HyDRA establishes that integer-valued, statically-verified, asynchronously-pipelineable programmability is achievable for microfluidic biochips — given a fluid substrate chosen with that abstraction in mind. The chip is realizable in standard PDMS within the ISO 22916 credit-card footprint; the ISA is small and SSA-disciplined; the formal model is finite and the verification tractable; the target assay class is expressible with margin. The natural continuation — fabricate, implement, measure, mechanize — is the subject of Chapter 14.

Chapter 14

Extensions and Future Work

This chapter identifies concrete continuations, grouped by character: an implementation roadmap that turns the specification into a working artifact (Section 14.1); architectural extensions that broaden the design within the existing framework (Section 14.2); and conceptual integrations that compose HyDRA with adjacent research (Section 14.3).

14.1 Implementation Roadmap

Four implementation tasks together would carry HyDRA from specification to deployable system, in roughly increasing order of effort.

14.1.1 Single-Tile Prototype

The most immediate continuation is a single-tile prototype: one register tile plus pump, I/O zone, and demux (no full 10-tile loop), fabricated from the standard multilayer-PDMS process — two SU-8 masters, PDMS molding, plasma-assisted lamination, Aquapel treatment. Such a prototype can execute all five microarchitectural primitives (Circ, SetR, SetMx, SetIO, Wait) and the basic ISA operations (create, destroy, split, merge), and would establish:

1. that the fluid system admits the predicted 4–14% coverage window in practice (direct measurement of fusion timescales and wall-film drainage);
2. reproducibility of γ within $\varepsilon_{\max} = 1\%$ per stroke;
3. cell-level PARK re-registration accuracy (high-speed imaging of slug-valve contact at precision velocity);
4. valve-safety precondition efficacy under deliberate stress-testing.

The prototype is a two-month fabrication-and-instrumentation effort using standard microfluidic laboratory equipment.

14.1.2 Reference Compiler

A reference compiler is a well-defined engineering task: parse textual ISA programs; SSA-verify (Chapter 10, Section 10.2); allocate registers to live slugs (Section 10.5); lower each ISA operation to microarchitectural primitives (Chapter 9); verify the lowered program against the eight invariants (Chapter 8); emit a timestamped valve-actuation stream. A prototype in a functional language (OCaml, Haskell) or a proof assistant (Rocq/Coq) is a few-thousand-line effort; the formal model is tight enough that the verification component transcribes directly to predicate-check routines.

14.1.3 Calibration Protocol

The per-chip Γ_{sat} calibration identified in Chapter 6 is the only measurement required to place a specific chip inside the predicted 4–14% window. Procedure: measure dynamic surface tension $\sigma(t)$ by pendant-drop tensiometry on the specific surfactant batch in HFE-7500 over 0.1-CMC to CMC concentrations; fit Langmuir or Frumkin isotherm; extract Γ_{sat} and the bulk concentration corresponding to the target coverage range; use as the surfactant dose. The procedure takes a working day with a standard pendant-drop tensiometer, once per surfactant batch.

14.1.4 Machine-Checked Correctness

The eight invariants are proved by pen-and-paper structural induction over the ISA operations and case analysis on mode transitions; the proofs use no unstated lemma and are routine for a formal-methods expert. Mechanization in a proof assistant (Coq, Lean, Isabelle, or equivalent) would produce a machine-checked certificate that the architecture is free of the classes of errors it claims to exclude. Plausible targets: formalization of the state (Σ, C, ν, F) as a dependently-typed record; definition of each ISA operation as a state transformation with explicit pre- and post-conditions; proof that each operation preserves all eight invariants; proof that the verification procedure is a decision procedure for program safety; and integration with a verified compiler, certifying that any accepted program is safe. Such a mechanization would place HyDRA alongside verified-compiler efforts (e.g. CompCert) in the programming-language formalization literature.

14.2 Architectural Extensions

Four extensions broaden the design within the existing layered specification.

14.2.1 Dual-Pump Variant

A variant with two pumps placed diametrically opposite on the bus would halve the worst-case per-pump active-path length entering the compliance calculation (relaxing $n_{\text{XC}} \leq 2$ to $n_{\text{XC}} \leq 4$ across the loop), halve PARK distance (reducing inter-checkpoint drift), and admit splitting the bus into two independent arcs during some operations (recovering bus-side parallelism), at the cost of doubled valve count, doubled pneumatic complexity, and pump synchronization. The ISA extension is minor — a pump-designator argument at the Circ level — and the formal model requires only a second active-path component with a synchronization constraint. The dual-pump variant is a concrete scaling option should practical experience with the single-pump specification identify concurrent-register-access density as a bottleneck.

14.2.2 Scaling and Demultiplexer Validation

$N = 10$ matches the target assay class; scaling is a pure parametric change since no invariant, ISA operation, or microarchitectural primitive depends on $N = 10$ specifically. A 20- or 40-tile chip within the ISO large-chip envelope (128×86 mm) would extend coverage to classes with larger antichain widths — parallel crystallization trials, combinatorial chemistry, replicated dose-response sweeps. The demultiplexer scales logarithmically (20 tiles require 8 bits for ~ 150 valves); the principal concrete gap is empirical validation of the $k = 4 \to 7$ extrapolation from Grover [17, 16], with Weaver pressure-gain stages [43] restoring signal margin. A standalone 7-bit demultiplexer test chip — no HyDRA fluidics, just the demux — measuring addressing error rate and actuation latency would close this gap in a two-week effort.

14.2.3 Data-Dependent Iteration

`repeat` provides compile-time unrolling of a fixed count; extending the ISA to data-dependent iteration (iteration count depending on a runtime readout) would cover titration-to-endpoint, feedback-controlled dilution, and adaptive-dose response. Two compromises preserve tractability:

- **Bounded-runtime loops.** The loop executes at most $N_{\max}$ iterations (a compile-time constant) with an early-exit `branch` inside the body. Expressible in the current ISA without extension.

- **Loop invariants.** The programmer supplies an invariant; the compiler verifies that one iteration preserves it. Verification becomes independent of iteration count. Requires programmer annotations and additional compiler machinery.

The former is a pragmatic engineering solution covering many realistic cases without architectural change; the latter is a formal-methods research direction.

14.2.4 Heterogeneous Tiles

Tiles may bind to the same module type (homogeneous) or different types (heterogeneous). Heterogeneity supports richer assay classes — e.g. one thermocycler tile, two incubators, one fluorometer, and six general-purpose tiles — at the cost of a register-allocation constraint forcing specific slugs onto specific tiles. The allocator extension is straightforward: the homogeneous case is a special case.

14.3 Conceptual Integrations

Two integrations compose HyDRA with adjacent research without requiring substrate-level changes.

14.3.1 Chemistry Type Systems

BioScript's chemistry type system [31] tracks reagent identity and safety properties at the ISA level. Integrating it would layer *chemical* safety (forbidden reagent combinations) on top of HyDRA's *substrate-level* safety (valve splits, register overflows), giving the programmer a unified compile-time safety envelope. The integration is conceptually clean: BioScript's types are annotations on the slug identifier, flowing through the SSA discipline as auxiliary data without affecting the lowering or the substrate-level invariants.

14.3.2 Additional Assay Classes

The target class of Chapter 4 is representative but not exhaustive. Compatible extensions, requiring no architectural change, include single-cell sequencing (each droplet contains a single cell; the substrate is compatible, but module types — cell lysis, barcoding, library preparation — require characterization) and organic-synthesis or materials applications (slug flow, pump quantum, and integer ISA apply to any miscible-phase chemistry with discrete reagent volumes; the abstractions are not specific to biochemistry).

Incompatible classes include digital droplet PCR (thousands of parallel droplets, poorly matched to a single-bus architecture; a different variant with an array of small pumps would be required) and electrophoretic separations (spatial concentration gradients are fundamentally incompatible with discrete slugs). Scope is a design decision: HyDRA commits to a specific substrate class and accepts that other classes require different architectures.

14.4 Conclusion

HyDRA is a complete architectural specification, but a specification is only as valuable as the artifacts built on it. A fabricated prototype, a reference compiler, a machine-checked formalization, and an extended assay class each represent a natural next step. The thesis puts the architectural foundation in place; the ecosystem that builds on it remains to be constructed.

Bibliography

[1] 3M Company. 3M Novec HFE-7500 engineered fluid: Product datasheet. Technical document, 3M, 2024.

[2] S. Alexander. Adsorption of chain molecules with a polar head: A scaling description. *J. Phys. France*, 38(8):983–987, 1977.

[3] Ahmed M. Amin, Mithuna Thottethodi, T. N. Vijaykumar, Steven Wereley, and Stephen C. Jacobson. Aquacore: A programmable architecture for microfluidics. In *Proc. ISCA*, pages 254–265, 2007.

[4] Vaishnavi Ananthanarayanan and William Thies. Biocoder: A programming language for standardizing and automating biology protocols. *J. Biol. Eng.*, 4:13, 2010.

[5] P. Aussillous and D. Quéré. Quick deposition of a fluid on the wall of a tube. *Phys. Fluids*, 12(10):2367–2371, 2000.

[6] Jean-Christophe Baret. Surfactants in droplet-based microfluidics. *Lab Chip*, 12(3):422–433, 2012.

[7] F. P. Bretherton. The motion of long bubbles in tubes. *J. Fluid Mech.*, 10(2):166–188, 1961.

[8] Krishnendu Chakrabarty, Richard B. Fair, and Jun Zeng. Design tools for digital microfluidic biochips: Toward functional diversification and more than Moore. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 29(7):1001–1017, 2010.

[9] H.-P. Chou, M. A. Unger, and S. R. Quake. A microfabricated rotary pump. *Biomedical Microdevices*, 3(4):323–330, 2001.

[10] Rémi Dangla, S. Cagri Kayi, and Charles N. Baroud. Droplet microfluidics driven by gradients of confinement. *Proc. Nat. Acad. Sci. USA*, 110(3):853–858, 2013.

[11] Pierre-Gilles de Gennes, Françoise Brochard-Wyart, and David Quéré. *Capillarity and Wetting Phenomena: Drops, Bubbles, Pearls, Waves*. Springer, 2004.

[12] Emulseo. FluSurf-C: Product data sheet. Technical document, Emulseo SAS, 2024.

[13] Thomas Gervais, Jamil El-Ali, Axel Günther, and Klavs F. Jensen. Flow-induced deformation of shallow microfluidic channels. *Lab Chip*, 6(4):500–507, 2006.

[14] Tristan Gilet and Sébastien van Loo. Multiple interactions between microfluidic droplets and on-chip pneumatic valves. *Microfluid. Nanofluid.*, 26:20, 2022.

[15] J. Goulpeau, D. Trouchet, A. Ajdari, and P. Tabeling. Experimental study and modeling of polydimethylsiloxane peristaltic micropumps. *J. Appl. Phys.*, 98(4):044914, 2005.

[16] William H. Grover, Robin H. C. Ivester, Erik C. Jensen, and Richard A. Mathies. Development and multiplexed control of latching pneumatic valves using microfluidic logical structures. *Lab Chip*,

6(5):623–631, 2006.

[17] William H. Grover, Alison M. Skelley, Chung N. Liu, Eric T. Lagally, and Richard A. Mathies. Monolithic membrane valves and diaphragm pumps for practical large-scale integration into glass microfluidic devices. *Sens. Actuators B: Chem.*, 89(3):315–323, 2003.

[18] Hao Gu, Michel H. G. Duits, and Frieder Mugele. Droplets formation and merging in two-phase flow microfluidics. *Int. J. Mol. Sci.*, 12(4):2572–2597, 2011.

[19] A. Günther and K. F. Jensen. Multiphase microfluidics: from flow characteristics to chemical and materials synthesis. *Lab on a Chip*, 6(12):1487–1503, 2006.

[20] C. L. Hansen, E. Skordalakes, J. M. Berger, and S. R. Quake. A robust and scalable microfluidic metering method that allows protein crystal growth by free interface diffusion. *Proceedings of the National Academy of Sciences*, 99(26):16531–16536, 2002.

[21] International Organization for Standardization. ISO 22916:2022 Microfluidic Devices — Interoperability Requirements for Dimensions, Connections and Initial Device Classification. ISO Standard, 2022.

[22] Jacob N. Israelachvili. *Intermolecular and Surface Forces*. Academic Press, 3rd edition, 2011.

[23] I. D. Johnston, D. K. McCluskey, C. K. L. Tan, and M. C. Tracey. Mechanical characterization of bulk Sylgard 184 for microfluidics and microengineering. *Journal of Micromechanics and Microengineering*, 24(3):035017, 2014.

[24] E. P. Kartalov, J. F. Zhong, A. Scherer, S. R. Quake, C. R. Taylor, and W. F. Anderson. High-throughput multi-antigen microfluidic fluorescence immunoassays. *BioTechniques*, 40(1):85–90, 2006.

[25] E. T. Lagally, I. Medintz, and R. A. Mathies. Single-molecule DNA amplification and analysis in an integrated microfluidic device. *Analytical Chemistry*, 73(3):565–570, 2001.

[26] Jessamine Ng Lee, Cheolmin Park, and George M. Whitesides. Solvent compatibility of poly(dimethylsiloxane)-based microfluidic devices. *Analytical Chemistry*, 75(23):6544–6554, 2003.

[27] Xavier Leroy. Formal verification of a realistic compiler. *Communications of the ACM*, 52(7):107–115, 2009.

[28] Linas Mazutis and Andrew D. Griffiths. Selective droplet coalescence using microfluidic systems. *Lab Chip*, 12(10):1800–1806, 2012.

[29] Jessica Melin and Stephen R. Quake. Microfluidic large-scale integration: The evolution of design rules for biological automation. *Annu. Rev. Biophys. Biomol. Struct.*, 36:213–231, 2007.

[30] Wajid Hassan Minhass, Paul Pop, Jan Madsen, and Felician Stefan Blaga. Architectural synthesis of flow-based microfluidic large-scale integration biochips. *Proc. Int. Conf. Compilers, Architectures and Synthesis for Embedded Systems (CASES)*, pages 181–190, 2012.

[31] Jason Ott, Tyson Loveless, Chris Curtis, Mohsen Lesani, and Philip Brisk. BioScript: Programming safe chemistry on laboratories-on-a-chip. *Proc. ACM Program. Lang.*, 2(OOPSLA):128:1–128:31, 2018.

[32] E. A. Ottesen, J. W. Hong, S. R. Quake, and J. R. Leadbetter. Microfluidic digital PCR enables multigene analysis of individual environmental bacteria. *Science*, 314(5804):1464–1467, 2006.

[33] Osborne Reynolds. On the theory of lubrication and its application to Mr. Beauchamp Tower's experiments, including an experimental determination of the viscosity of olive oil. *Philosophical Transactions of the Royal Society of London*, 177:157–234, 1886.

[34] Birte Riechers, Florine Maes, Elias Akoury, Benoît Semin, Philipp Gruner, and Jean-Christophe Baret. Surfactant adsorption kinetics in microfluidics. *Proc. Nat. Acad. Sci. USA*, 113(41):11465–11470, 2016.

[35] A. Sharma and E. Ruckenstein. Stability, critical thickness, and the time of rupture of thinning foam and emulsion films. *Langmuir*, 3(5):760–768, 1987.

[36] V. Srinivasan, V. K. Pamula, and R. B. Fair. An integrated digital microfluidic lab-on-a-chip for clinical diagnostics on human physiological fluids. *Lab on a Chip*, 4(4):310–315, 2004.

[37] Abraham D. Stroock, Stephan K. W. Dertinger, Armand Ajdari, Igor Mezić, Howard A. Stone, and George M. Whitesides. Chaotic mixer for microchannels. *Science*, 295(5555):647–651, 2002.

[38] D. Tabor and R. H. S. Winterton. The direct measurement of normal and retarded van der Waals forces. *Proceedings of the Royal Society of London A*, 312(1511):435–450, 1969.

[39] Georgi Tanev, Winnie E. Svendsen, and Jan Madsen. BiowareCFP: An application-agnostic modular reconfigurable cyber-fluidic platform. *Micromachines*, 13(2):249, 2022.

[40] William Thies, John Paul Urbanski, Todd Thorsen, and Saman Amarasinghe. Abstraction layers for scalable microfluidic biocomputing. *Natural Computing*, 7(2):255–275, 2008.

[41] Todd Thorsen, Sebastian J. Maerkl, and Stephen R. Quake. Microfluidic large-scale integration. *Science*, 298(5593):580–584, 2002.

[42] Marc A. Unger, Hou-Pu Chou, Todd Thorsen, Axel Scherer, and Stephen R. Quake. Monolithic microfabricated valves and pumps by multilayer soft lithography. *Science*, 288(5463):113–116, 2000.

[43] James A. Weaver, Jessica Melin, Don Stark, Stephen R. Quake, and Mark A. Horowitz. Static control logic for microfluidic devices using pressure-gain valves. *Nat. Phys.*, 6(3):218–223, 2010.

[44] George M. Whitesides. The origins and the future of microfluidics. *Nature*, 442(7101):368–373, 2006.

[45] Max Willsey, Ashley P. Stephenson, Chris Takahashi, et al. Puddle: A dynamic, error-correcting, full-stack microfluidics platform. In *Proc. ASPLOS*, pages 183–197, 2019.

[46] Harris Wong, C. J. Radke, and S. Morris. The motion of long bubbles in polygonal capillaries. part 1. thin films. *Journal of Fluid Mechanics*, 292:71–94, 1995.

Publications and Communications

A condensed version of this thesis has been submitted for peer review:

- Aarush Kumbhakern and Prof. Bhargab B. Bhattacharya, "HyDRA: A Slug-Flow Microfluidic Architecture and Instruction Set," submitted to the 2026 IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2026), Kolkata, India, July 2026. [Under review.]