VLSI Design and Circuits for Multiple Supply Domains

Bryan Ngo

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2024-126
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-126.html

May 17, 2024

VLSI Design and Circuits for Multiple Supply Domains

by Bryan Ngo

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Borivoje Nikolić
Research Advisor

May 16, 2024.
(Date)

* * * * * * *

Sophia Shao

Professor Sophia Shao
Second Reader

5/17/2024
(Date)

Abstract

VLSI Design and Circuits for Multiple Supply Domains

by

Bryan Ngo

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Borivoje Nikolić, Chair

Professor Sophia Shao, Co-chair

The rapid proliferation of mobile computing introduces several design challenges for a modern integrated circuit, such as power usage. A modern integrated circuit must be able to efficiently manage its power supply. There exists two ways of approaching the problem. The first is that of infrastructure, or creating an environment in which power-efficient designs can be easily conveyed and automatically inferred. The second is that of design, or creating circuits to facilitate power-efficient operation. This work approaches this problem from the perspective of VLSI infrastructure, as well as on-chip integrated circuits.

The first part of this work describes the implementation of multiple power domains in an open-source VLSI flow manager, HAMMER. We demonstrate an API that allows a user to specify arbitrary physical domains on chip, and generates a power format that can be further used to verify the domain specification. This API is able to be translated into a power intent that can be ingested by EDA tools and displayed in a real VLSI flow.

The second part of this work describes the design and implementation of a digitally-controlled low-dropout regulator (LDO). The LDO serves as a testbed for design space exploration both in the analog domain and the digital domain. We demonstrate a design with an integral control logic scheme achieving 144 mV dropout voltage, 200 ns settling time, and 99.6 % current efficiency, competitive with previous work.

To my family, who made all things possible.

Contents

List of Figures

List of Tables

List of Listings

Acknowledgments

This thesis would not be possible with the contributions of many people, technical or otherwise. First and foremost, I would like to express my greatest thanks to Professor Borivoje Nikolić, who trusted me to take on this research project and has guided me throughout the year. I would also like to thank Apple for awarding me the Masters Scholarship in Integrated Systems and funding my year.

My utmost gratitude goes out to Harrison Liew, Daniel Grubb, Ken Ho, Nayiri Krysztofowicz, and the rest of HAMMER for introducing me to this field years ago; I would not be the researcher I am today without their guidance and influence. Additional thanks goes out to all my colleagues in SLICE Lab and the Berkeley Wireless Research Center for their feedback and collaboration.

My 5 years here at UC Berkeley would not be nearly as enjoyable without the many friends I've made along the way. Thank you all for giving me a community I could celebrate with at my highest points and rely on at my lowest points.

Finally, I thank my family for supporting me through thick and thin for the past 23 years. They have made sure I was able to achieve my goals, and words cannot express how gracious I am for their presence.

Chapter 1

Multiple Power Domains in VLSI

1.1 Motivation

Very large scale integration (VLSI) is the modern practice of designing digital chips with billions of transistors on a single die. This allows the designer to create complete systems on a chip (SoCs) with multiple functions. To facilitate this complexity, there are three facets of VLSI: the design, the technology, and the tool.

The design is the desired functionality of the SoC, usually some register-transfer level description of the block. The technology is the process that the design will be fabricated in. This encapsulates details such as the standard cells that a synthesis tool would employ, the design rules required to reliably fabricate the physical layout, or the types of usable metal connections. These are packaged into a process design kit (PDK). For example, all circuits in this paper use the open-source Skywater 130 nm CMOS process for reproducibility. [1] Finally, the tool takes both the design and attempts to convert the relatively abstract description into a transistor-level layout that can be cleaned and, eventually, passed off to a foundry. This is done in several steps. The first is that of *synthesis*, or the process of turning RTL into a netlist of logical expressions. Specifically, they are expressed using the PDK's standard cell library, which gives a list of fundamental logical functions to choose from. This netlist can then be passed on to the *place-and-route* (PAR or P&R) step. The PAR tool will take the previous netlist, instantiate all the necessary standard cells into their physical descriptions, then route them together using the process' allowed metal layers. At this point, the design can be passed through a design rule check (DRC) to check obedience against the process' design rules, or a layout versus schematic (LVS) check to ensure that the electrical functionality of the design is satisfied.

The sheer variance in the types of designs, technologies, and tools lend to the vast complexity of the overall VLSI process. This motivates the development of VLSI flow tools to manage this complexity in an idiomatic fashion, allowing IC designers to create portable and scalable designs. Highly Agile Masks Made Effortlessly from RTL (HAMMER/Hammer) is one such tool [2].

1.2 Hammer Usage

The traditional method of "pushing" a digital design through a standard VLSI flow involves heavy use of scripting languages like Tcl to interact using a list of commands that a given tool, such as Cadence's Genus logic synthesis tool. This is a highly non-portable solution for several reasons. Each new design requires the creation of a wholly new script, since the constraints can differ drastically. Furthermore, the existence of multiple different technologies and associated ramifications compounds the issue. Hammer aims to decouple the concerns of the design, the tool, and the technology from each other. This nominally allows a user to change a design, start using a different tool, or attempt integration in a different technology altogether without having to propagate the change.

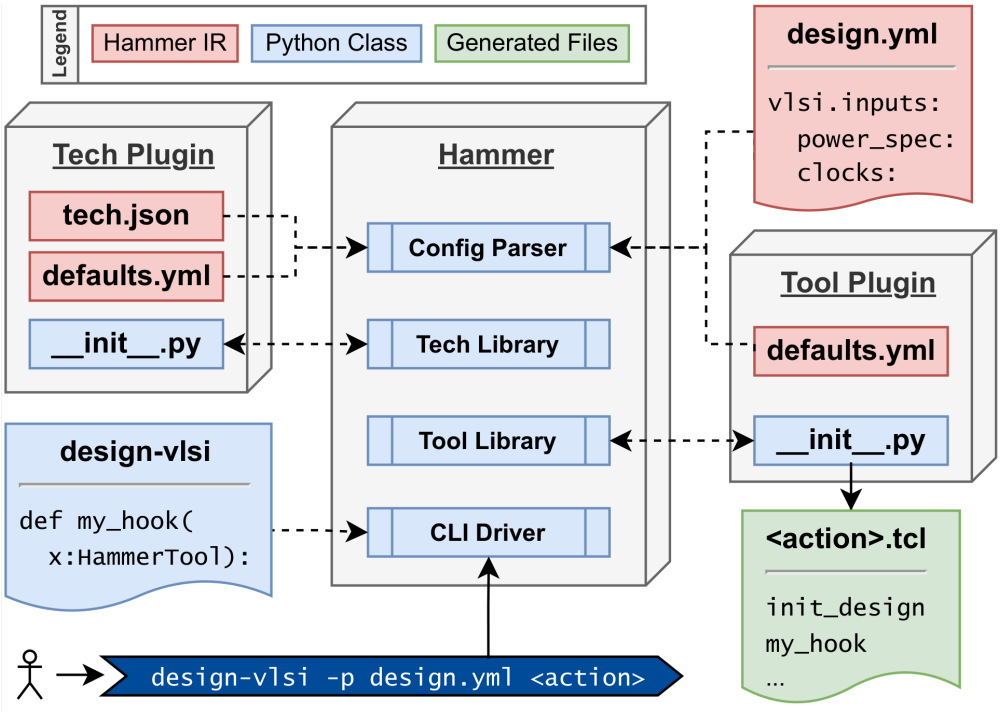

Figure 1.1: Architecture of Hammer.

As shown in Figure 1.1, a user can configure the entirety of the design's VLSI toolchain through Hammer IR, which is nothing more than one or more YAML markup files. Hammer takes this information, along with an advanced feature called *hooks*, and emits the same Tcl script that would have been handwritten that the tool recognizes and accepts. Hammer expects the contents of these "config files" to adhere to a schema that defines certain properties. The keys and values from these config files are ingested into a database that Hammer uses every time the user wants to perform an "action", such as synthesis, place-and-route,

etc. These can include but are not limited to the desired tools to perform a given action with, the specifications for the desired clock frequency, or the physical locations for desired sub-blocks on the die.

```
# Specify clock signals
vlsi.inputs.clocks: [
  {name: "clk", period: "10ns", uncertainty: "0.1ns"}
]

sim.inputs:
  defines: ["CLOCK_PERIOD=10"]
  defines_meta: "append"

# Placement Constraints
vlsi.inputs.placement_constraints:
  - path: "gcd"
    type: toplevel
    x: 0
    y: 0
    width: 1000
    height: 1000
    margins:
      left: 10
      right: 10
      top: 10
      bottom: 10

vlsi.inputs.delays: [
  {name: "reset",           clock: "clk", delay: "1", direction: "input"},
  {name: "operands_bits_A", clock: "clk", delay: "1", direction: "input"},
  {name: "operands_bits_B", clock: "clk", delay: "1", direction: "input"},
  {name: "operands_val",    clock: "clk", delay: "1", direction: "input"},
  {name: "operands_rdy",    clock: "clk", delay: "1", direction: "output"},
  {name: "result_bits_data",clock: "clk", delay: "1", direction: "output"},
  {name: "result_val",      clock: "clk", delay: "1", direction: "output"},
  {name: "result_rdy",      clock: "clk", delay: "1", direction: "input"}
]
```

Listing 1: A minimal example of a GCD module config file.

As shown in Listing 1, we can configure properties such as the clock uncertainty or the de-

sired block width and height. Of particular interest to this research is the `placement_constraints` key. This key defines a standard `PlacementConstraint` data structure in Hammer. There are various type of `PlacementConstraint`s available in Hammer, such as the actual placement and dimensions of a block, or the location of an obstruction to restrict the PAR tool in that particular area.

1.3 Power Domains in VLSI

Digital designs require precise specification of the power being supplied to each block. This is done via a specification called a *power format*. There are two power formats currently in widespread use: the Universal Power Format (UPF) [3] and the Common Power Format (CPF). [4] Both of these formats have the equivalent expressiveness for a *power intent*, or the abstract specification for how a user wants to specify the power grid distribution on their chip. Since CPF is the primary accepted format for tools like Cadence Innovus, this paper will primarily focus on this format.

To simplify, a power format consists of specifying *power domains*, which defines physical areas on-chip to be powered at specified voltages. Then, *power nets* are specified as electrical connections. Then, one must specify the interactions between adjacent power domains, such as whether to add level shifters to raise the voltage or to add isolation to prevent data from being corrupted or misreceived from one domain to the other.

```
# This script is generated by HAMMER
set_cpf_version 1.0e
set_hierarchy_separator /
set_design z1top.xdc
create_power_nets -nets AO -voltage 0.85
create_power_nets -nets A -voltage 0.45
create_power_nets -nets B -voltage 0.5
create_ground_nets -nets { VSS }
create_power_domain -name AO -default
create_power_domain -name A
create_power_domain -name B
update_power_domain -name AO -primary_power_net AO -primary_ground_net VSS
update_power_domain -name A -primary_power_net A -primary_ground_net VSS
update_power_domain -name B -primary_power_net B -primary_ground_net VSS
create_global_connection -domain AO -net AO -pins [list VDD]
create_global_connection -domain A -net A -pins [list A]
create_global_connection -domain B -net B -pins [list B]
create_global_connection -domain AO -net VSS -pins [list VSS]
create_level_shifter_rule -name AO_LS -from AO -to [list A B]
create_nominal_condition -name nominal -voltage 0.85
create_power_mode -name aon -default -domain_conditions {AO@nominal}
create_nominal_condition -name AO_condition -voltage 0.85
create_power_mode -name AO -domain_conditions {AO@AO_condition}
create_nominal_condition -name A_condition -voltage 0.45
create_power_mode -name A -domain_conditions {A@A_condition}
create_nominal_condition -name B_condition -voltage 0.5
create_power_mode -name B -domain_conditions {B@B_condition}
end_design
```

Listing 2: An example of a CPF with three distinct power domains.

Going line-by-line through Listing 2, we can see that three domains, `AO` (standard notation for a default "always on" domain), `A`, and `B`. Next, the power nets are created with a specified voltage. Note that the power nets sharing the same name as the power domains is for the sake of convenience; the domains themselves can be named arbitrarily as long as the net connections are consistent. The relevant connections between the net and the pins of the design are made as well. Then, we specify that level shifters are to be instantiated to shift between the `AO` domain to both `A` and `B`, but notably not *between* `A` and `B`. Finally, power conditions are specified to ensure that the power nets remain at a stable voltage throughout operation. A similarly expressive example can be demonstrated for the UPF format.

1.4 Changes to Power Domain Implementation

The existing power domain implementation in Hammer only supports one power net and one ground net, as shown in Listing 3

```
vlsi.inputs:
    power: [{name: "VDD", pins: ["VDD"]}]
    ground: [{name: "VSS", pins: ["VSS"]}]
    VDD: "0.85 V"
    GND: "0 V"
```

Listing 3: The existing power domain section of Hammer's default config file.

As is shown, only one power net and associated domain are allowed in current designs. In order to support multiple domains, the existing configuration file schema was modified through the course of this project. Each power/ground net dictionary now supports a `domain` key, allowing the user to connect any net to any domain.

Furthermore, a new power domain placement constraint can be specified. This can be done either as a standalone constraint or baked into an existing one, both examples of which are shown in Listing 4.

```yaml
placement_constraints:
  - path: "gcd"
    type: toplevel
    x: 0
    y: 0
    width: 1000
    height: 1000
    margins: {left: 10, right: 10, top: 10, bottom: 10}
    power_domain: "AO"

  - type: powerdomain
    path: gcd/A
    x: 0
    y: 0
    width: 500
    height: 500
    power_domain: "A"

  - type: powerdomain
    path: gcd/B
    x: 500
    y: 0
    width: 500
    height: 500
    power_domain: "B"
```

Listing 4: Example of new power domain placement constraints.

This instantiates three power domains in the designated coordinates and areas as shown in Figure 1.2.

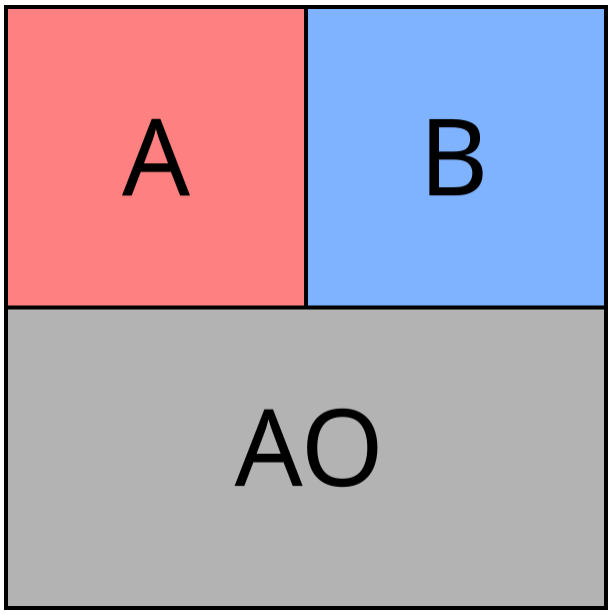

Figure 1.2: The desired power domain placement.

Then, when specifying nets as in Listing 3, a new `domain` key must be specified to connect the domain to a net. These changes combined give Hammer sufficient information to generate more flexible and robust power formats such as in Listing 2. The desired intent is then reflected by a tool like Cadence Innovus, as seen in Figure 1.3

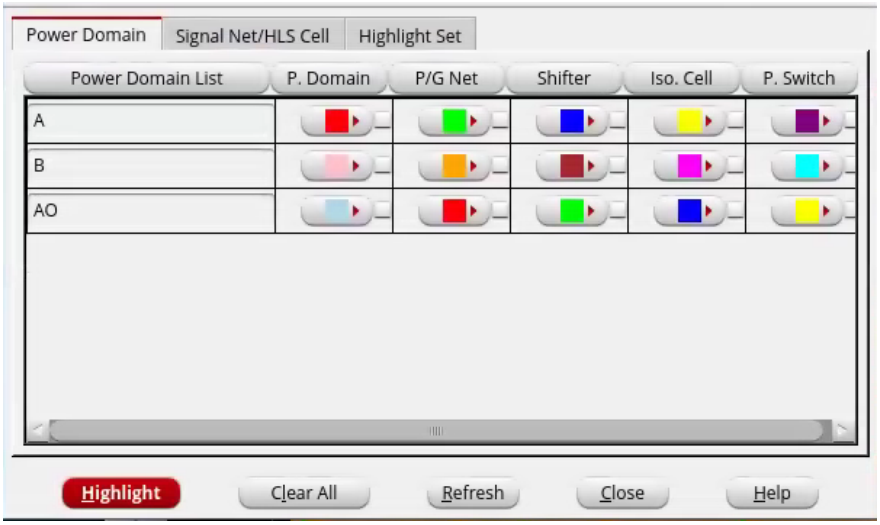

Figure 1.3: Multiple domains being ingested from the CPF in Cadence Innovus.

Many core Hammer functions for tasks such as querying all nets and their voltages were written with the assumption of a single power domain in mind. These core functionalities were rewritten to scale with many power domains and net connections. These downstream changes allow for a UPF and CPF generator to support this novel configuration file API. The overall changes to Hammer's core functionality can be seen in [5].

A multiple power domain architecture also requires the use of special standard cells to enable functionality such as converting from a lower voltage to a higher one by way of a level shifter, or enforcing isolation between two power nets. Each process technology (e.g. Skywater 130, Intel 16) has its own library, so these cells must be declared on a tech-by-tech basis. An example of how this is done is in Listing 5

```
"special_cells": [
    {
      "cell_type": "levelshifter",
      "name": [
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_4",
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_hl_isowell_tap_1",
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_hl_isowell_tap_2",
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_hl_isowell_tap_4",
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_tap_1",
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_tap_2",
        "sky130_fd_sc_hd__lpflow_lsbuf_lh_isowell_tap_4"
      ]
    },
    {
      "cell_type": "isolation",
      "name": [
        "sky130_fd_sc_hd__lpflow_inputiso0n_1",
        "sky130_fd_sc_hd__lpflow_inputiso0p_1",
        "sky130_fd_sc_hd__lpflow_inputiso1n_1",
        "sky130_fd_sc_hd__lpflow_inputiso1p_1",
        "sky130_fd_sc_hd__lpflow_inputisolatch_1",
        "sky130_fd_sc_hd__lpflow_isobufsrc_1",
        "sky130_fd_sc_hd__lpflow_isobufsrc_2",
        "sky130_fd_sc_hd__lpflow_isobufsrc_4",
        "sky130_fd_sc_hd__lpflow_isobufsrc_8",
        "sky130_fd_sc_hd__lpflow_isobufsrckapwr_16"
      ]
    }
]
```

Listing 5: A modified excerpt from Hammer's Skywater 130 special cell JSON, with multiple power domain cells declared.

Chapter 2

Design of a Digital LDO in 130 nm CMOS

2.1 Motivation

A low-dropout regulator (LDO) is a power management circuit used frequently in analog and mixed-signal design to maintain a constant supply voltage. The low dropout refers to a key specification known as dropout voltage, which is defined as the difference between the LDO's supply voltage and the maximum voltage the LDO can regulate against. A typical analog LDO consists of an operational amplifier (op-amp) connected in feedback with a MOSFET. The op-amp amplifies the error between the reference voltage and the voltage measured across an arbitrary load. The output voltage controls the gate voltage of a MOSFET, usually P-type, which drives the necessary current to track the reference voltage as needed by the load.

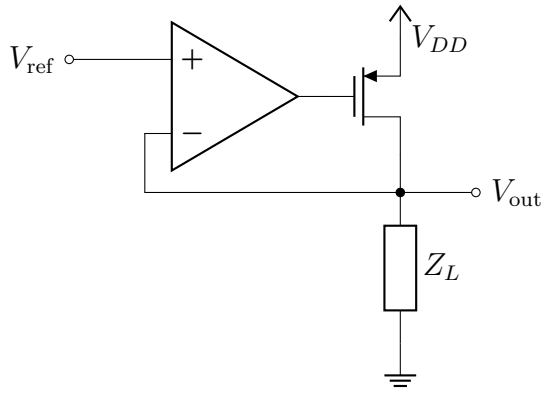

Figure 2.1: Typical analog LDO topology.

This topology offers various benefits, such as high bandwidth, and insignificant ripple [6]. However, the design is limited by the PMOS slew rate and the operating range of the PMOS. This motivates the design of the *digital* LDO, which in its prototypical form replaces the op-amp with an ADC (a comparator being the 1-bit special case of) feeding into digital logic, which outputs an N-bit bitmask that switches an array of N MOSFETs.

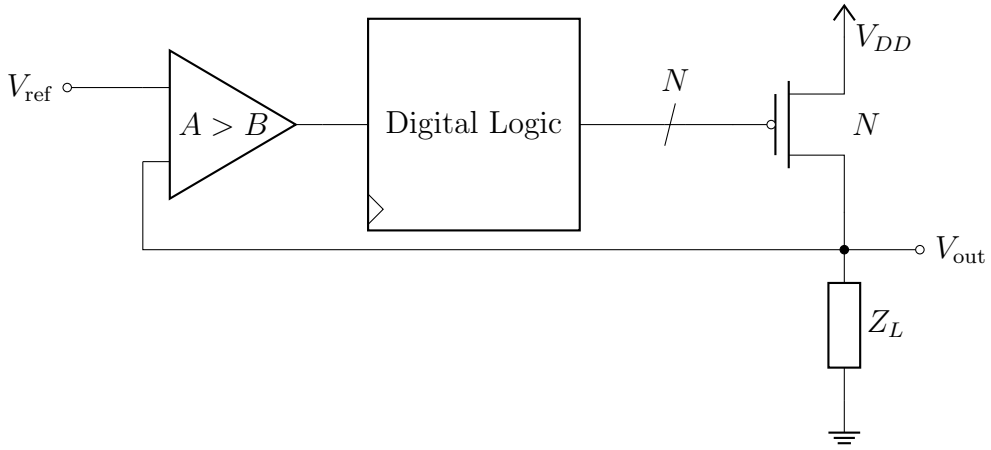

Figure 2.2: Typical digital LDO topology.

This topology can be desirable since the digital logic can consist of any arbitrary control logic, so the design is more scalable. Furthermore, the maximum load current and resolution is limited only by minimum PMOS size and area constraints. However, this design is limited by a low bandwidth and output ripple [6]. This work analyzes a particular implementation of a digitally-controlled LDO.

2.2 Components

Comparator

The comparator consists of a StrongARM latch followed by a digital D latch, as described in Figure 2.3.

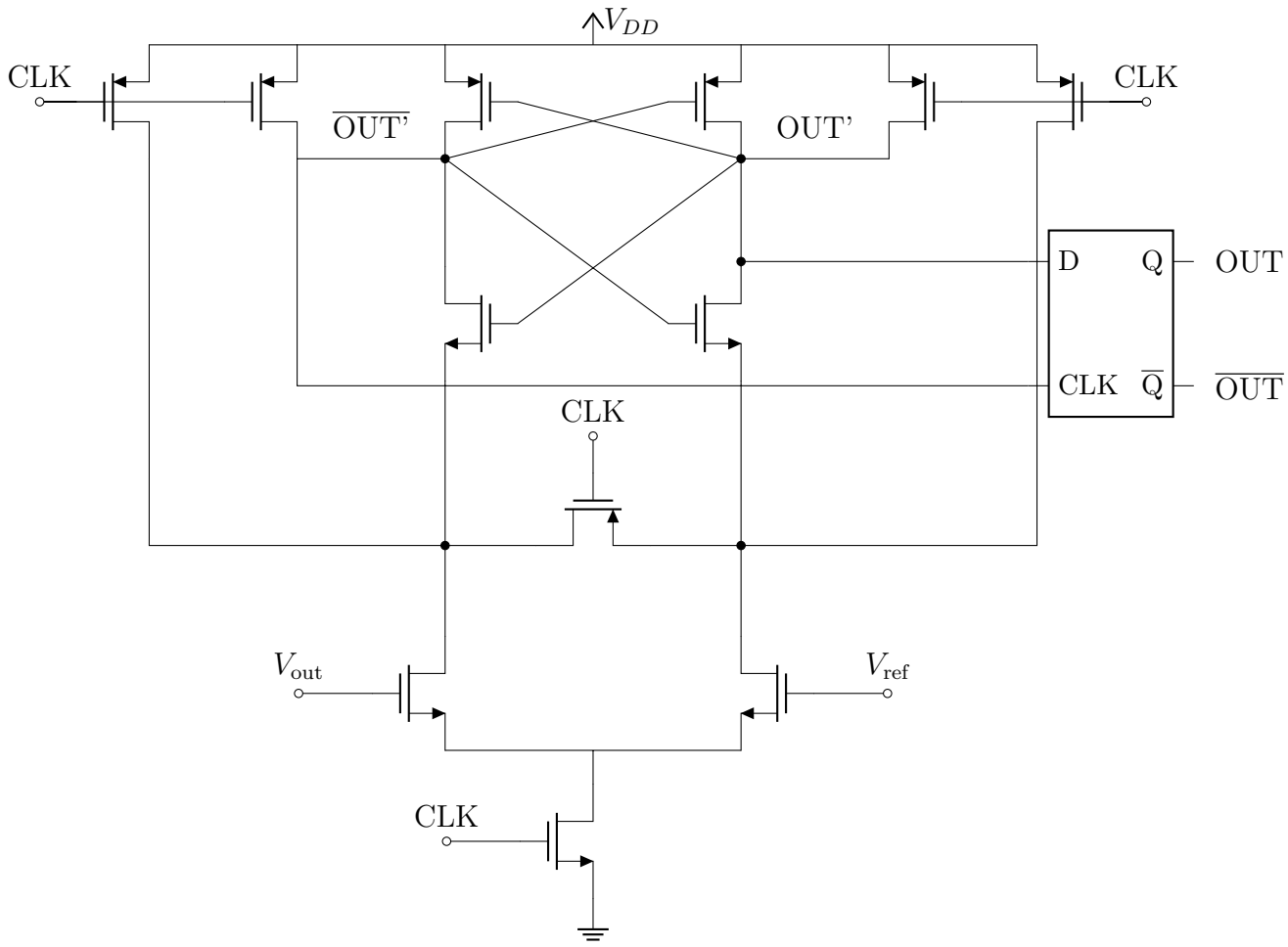

Figure 2.3: StrongARM latch with D latch comparator.

The StrongARM latch topology is drawn from [7]. The StrongARM latch and the variant form operates in two stages. At the rising edge of a clock, the latch effectively performs the operation $V_{\text{out}} > V_{\text{ref}}$ and reflects the result and its complement at OUT' and $\overline{\text{OUT'}}$, respectively. In between stages, the latch is in the *regeneration* stage, at which the comparator readout is invalid. The D latch output stage, as prescribed in [8], ensures sharp transitions and prevents ripple at the output. A D latch, as opposed to an SR latch in [8], is used to guard against the illegal state of the output and its complement being equal. The design has a $t_{C\to Q} = 1.5\,\text{ns}$, so this skew must be cancelled out when passing the clock onto the digital logic stage.

Digital Logic

The digital logic block determines how the LDO responds to error in the voltage difference at the load versus the reference voltage. The logic implemented is a form of integral control, represented by the difference equation

$$B[n] = B[n-1] + e[n] \tag{2.1}$$

where $e[n] \in \{-1, 1\}$ as the output of the comparator; note that in reality, the logical -1 is actually a voltage level of 0. We use the convention of a positive difference meaning that more current needs to be sourced from the pFET array, and vice versa with a negative difference. This means that when connecting the comparator output to the digital logic input, we use the complementary output. At every timestep, a bitmask counter is updated with the current error, so longer a positive error persists (implying a high current requirement) leads to more pFETs being switched.

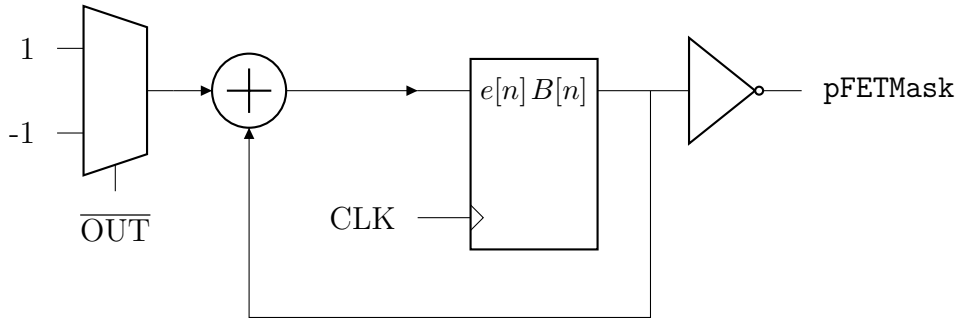

Figure 2.4: Digital logic system diagram.

The final output's bits are flipped since switching a pFET with its source connected to the voltage supply V_{DD} requires that the gate voltage be off for it to source current.

```
class DigitalLDOLogic(numPFET: Int) extends Module {
  val io = IO(new Bundle {
    val in = Input(Bool())
    val out = Output(UInt(numPFET.W))
  })

  val pFETMask = RegInit(0.U(numPFET.W))
  when (io.in && pFETMask < ~0.U(numPFET.W)) {
    pFETMask := pFETMask + 1.U
  }.elsewhen(!io.in && pFETMask > 0.U) {
    pFETMask := pFETMask - 1.U
  }
  io.out := ~pFETMask
}
```

Listing 6: Chisel code for LDO digital logic.

The digital logic was implemented in Chisel, a hardware description language developed at Berkeley to create parameterizable RTL generators [9]. This logic can be extensible for any number of pFETs in the array. One caveat not shown in Figure 2.4 is the counter saturation logic to prevent overflow when the current required exceeds the maximum or minimum current that the pFET array can provide. This Chisel RTL can then be transpiled to industry-standard Verilog by using the CIRCT compiler.

```
module DigitalLDOLogic(
  input        clock,
               reset,
               io_in,
  output [5:0] io_out
);

  reg [5:0] pFETMask;
  always @(posedge clock) begin
    if (reset)
      pFETMask <= 6'h0;
    else if (io_in & pFETMask != 6'h3F)
      pFETMask <= pFETMask + 6'h1;
    else if (~io_in & (|pFETMask))
      pFETMask <= pFETMask - 6'h1;
  end // always @(posedge)
  assign io_out = ~pFETMask;
endmodule
```

Listing 7: Transpiled Verilog for LDO digital logic.

Then, the Verilog RTL is taken through synthesis and PAR using Hammer to generate a Verilog file with the entire module defined using Skywater 130 standard cells. The command line tool `v2lvs` is used to convert the Verilog into a SPICE-compatible netlist. This netlist can be imported into an analog simulation tool such as Cadence Virtuoso ADE to simulate its performance.

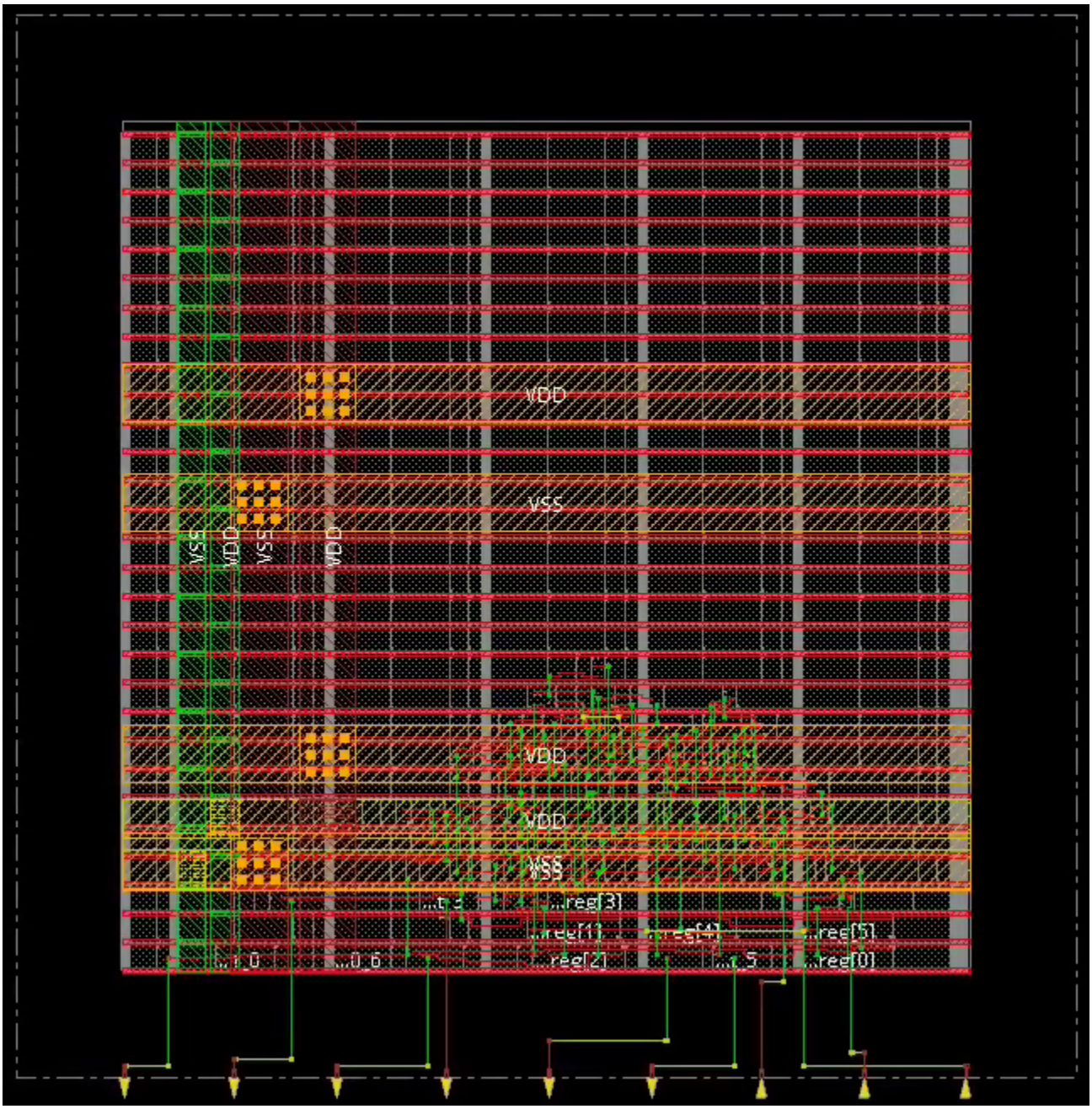

Figure 2.5: Post-PAR results of LDO digital logic.

One of the benefits of using a digitally-controlled LDO is the freedom to insert arbitrary control logic in the feedback loop, making the design's performance flexible and reiterable. However, this is coupled with the design of the pFET array, as will be discussed. The approach demonstrated in [7] uses bit shifting to switch 2^N equally-sized pFETs, as opposed

to this work that uses addition to switch N power-of-2 sized pFETs.

Another architecture attempted is a rendering of successive approximation, a version of which was used in [10].

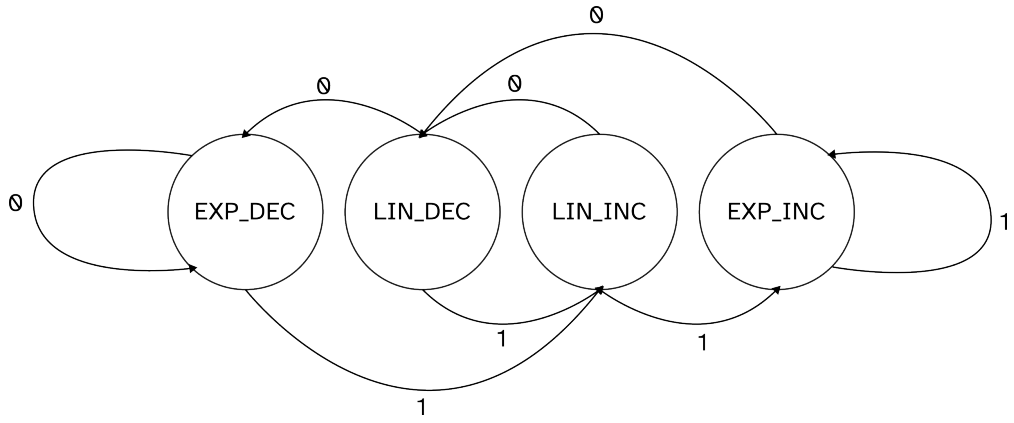

Figure 2.6: Finite state machine of LDO successive approximation.

The above state machine switches between exponential increases in sourced current (i.e. left shifting tbe bitmask), linear increases (i.e. adding to the bitmask), linear decreases (i.e. subtracting from the bitmask), and exponential decreases (i.e. right shifting the bitmask), depending on accumulated error. In short, the LDO will exponentially increase/decrease the current sourced until the first overshoot/undershoot, at which point it linearly backs off the current sourced until settling into a limit cycle. Figure 2.6 demonstrates the most basic version of this algorithm; the design can be extended to any number of intermediate states depending on the desired sensitivity and number of consecutive positive or negative errors detected until switching to exponential mode. Ideally, this control law should provide $O(N)$ settling time in terms of the bit resolution of the pFET array as opposed to $O(2^N)$ settling time for an integral control law. However, when integrated, the algorithm demonstrated extreme overshoot/undershoot even with an intermediate state, as corroborated in [10] and verified in Figure 2.9.

MOSFET Array

The final stage in a digitally-controlled LDO is the switched MOSFET array. In both analog and traditional digital LDOs, this power stage consists of P-type MOSFETs [6], [11]. The pFET array largely sets the maximum current the LDO is able to source as well as the resolution, or minimum current step, that the LDO can provide. Note that the linearity with respect to the source-gate voltage V_{SG} is not an issue since we are only driving the gate at either V_{DD} or 0 V. This relies on the linearity of the pFETs with respect to their aspect ratio, namely the width. A model of a pFET in saturation mode accounting for short-channel

effects (as would be seen in Skywater 130) presented in [12] expresses the drain current as

$$I_{DS} = \frac{W}{L}\frac{\mu C_{ox} E_c L}{2}\frac{(V_{SG} - V_{th})^2}{V_{SG} - V_{th} + E_c L} \propto W \tag{2.2}$$

This can be verified with a quick characterization of the pFETs. Ranging from $W = 500\,\text{nm}$ to $17.6\,\mu\text{m}$ with a constant $L = 300\,\text{nm}$, we can see that this is the case for Skywater 130 standard cell pFETs.

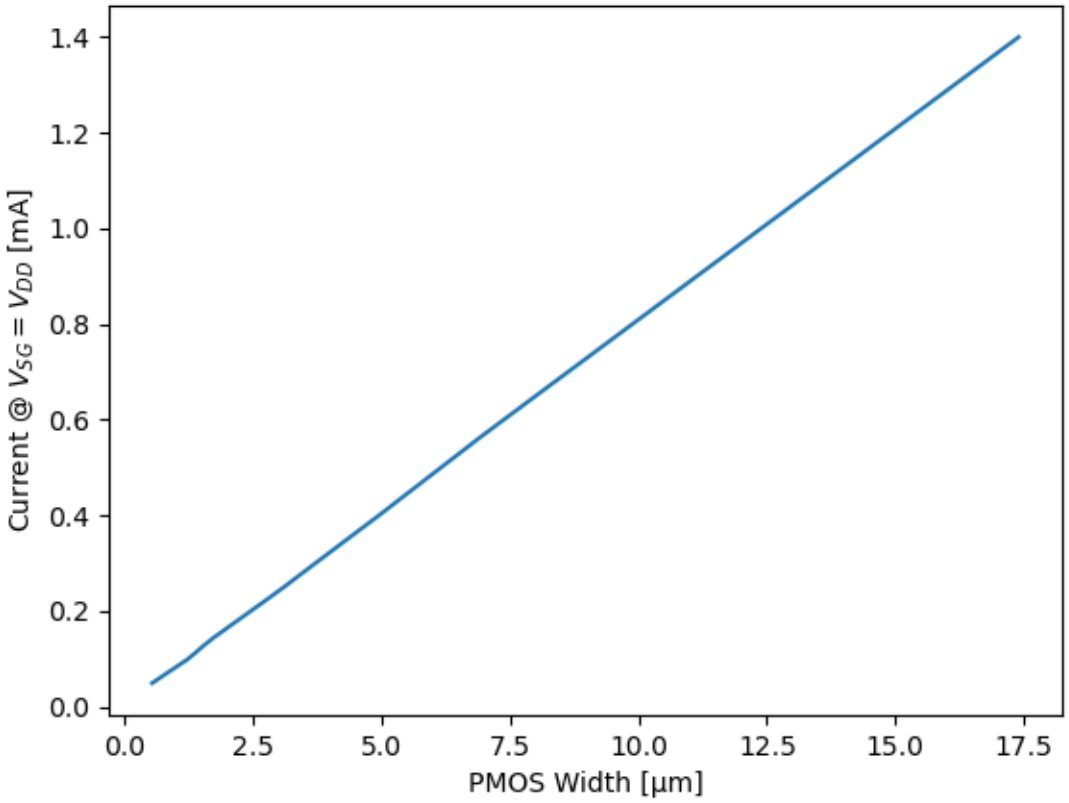

Figure 2.7: Current sourced at $V_{SG} = V_{DD}$ with respect to pFET width.

If the digital logic bitmask is intended to drive 2^N pFETs via shifting, then all of the pFETs can be equally sized and thus optimized for resolution. This has the benefit of not relying on linear width dependence and having the limit cycle limited by the current resolution. However, the number of pFETs scales exponentially with respect to bitwidth, making the design potentially difficult to scale. On the other hand, incrementing the bitmask

via addition and subtraction to drive N pFETs sized such that each one is double the width of the previous accomplishes the same goal. However, the limit cycle can now be as large as the largest pFET's current sourcing capability, in the case that the bitmask oscillates between `0b0111` and `0b1000` in the case of a 4-bit pFET array. This work chooses the N-bit architecture for ease of implementation. A 6-bit pFET array is implemented, with the smallest aspect ratio being $\frac{W}{L} = \frac{550\,\text{nm}}{300\,\text{nm}}$, with the width being the minimum allowable width for Skywater 130 standard pFETS. This allows for a maximum current of approximately 2.8 mA and a resolution of 50 µA, giving significant current range to the LDO. Notably, there exists a trade-off when using a linear search pFET array between that of settling time and resolution. The largest current step the digital LDO can make at any given clock cycle is the resolution $I_{DS,\text{LSB}}$. This means that the settling time to overcome an absolute difference ΔI_{load} can be expressed as

$$t_{\text{settle}} = \frac{\Delta I_{\text{load}}}{I_{DS,\text{LSB}} \cdot f_{\text{clk}}} \tag{2.3}$$

This is an inherent limitation of any linear search algorithm, and can be overcome with a properly-implemented binary search algorithm like successive approximation.

2.3 Results

Two architectures were implemented: basic integral control and a two-edge successive approximation. The digital LDO was implemented using the Skywater 130 nm CMOS process and is provided with a 100 MHz clock.

The transient behavior of the digital LDO was tested using a 1.8 V supply, a 0.8 V reference voltage to track, a constant 800 µA current load, and a 100 fF capacitive load.

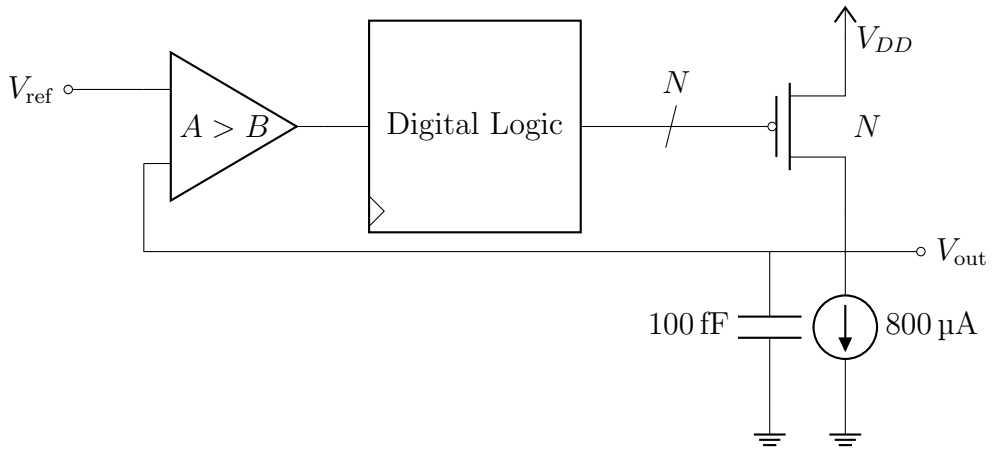

Figure 2.8: Digital LDO testing infrastructure.

To demonstrate the infeasibility of a successive approximation-like algorithm without augmentation, a modified version of Figure 2.6 with an additional intermediate linear state was implemented in the same transient test.

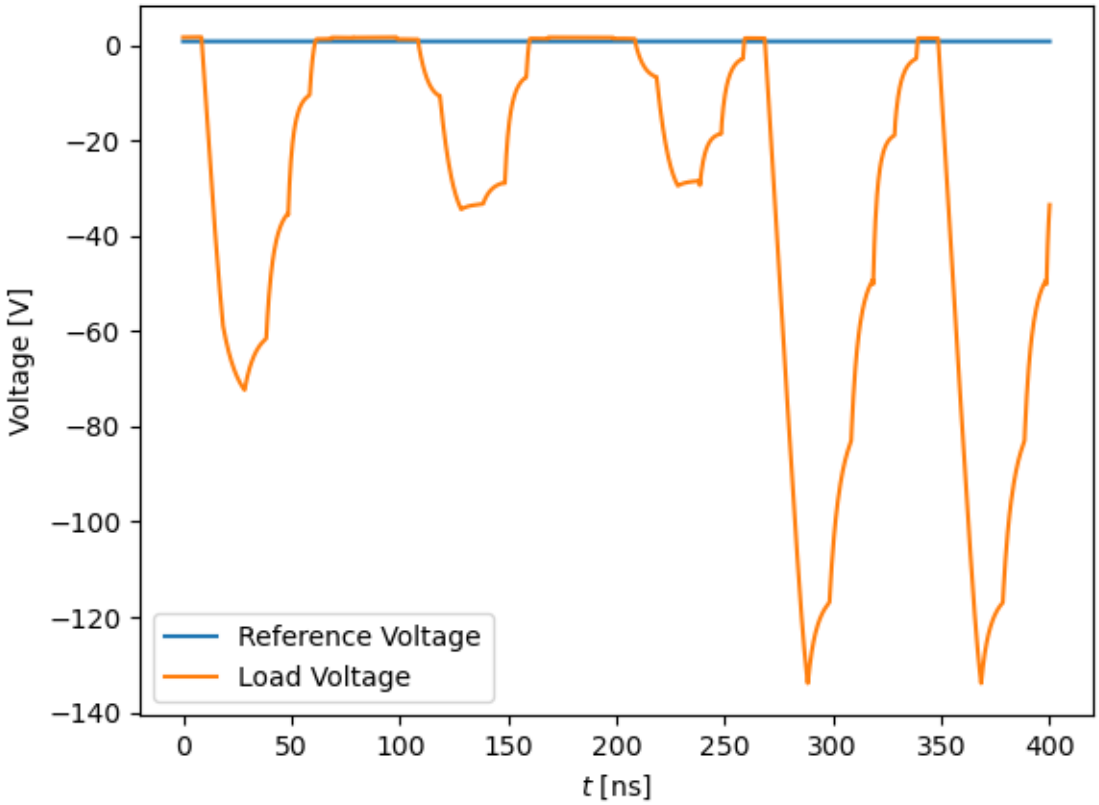

Figure 2.9: Transient response with successive approximation logic.

On one hand, the LDO approaches the target voltage much faster at approximately 60 ns, faster than was theoretically predicted with Equation 2.3. However, the criticisms levied against successive approximation LDOs in [7] appear in the transient response, as the finite state machine reacts sharply as the LDO approaches the reference voltage. For this reason, the rest of the work will analyze the integral control LDO.

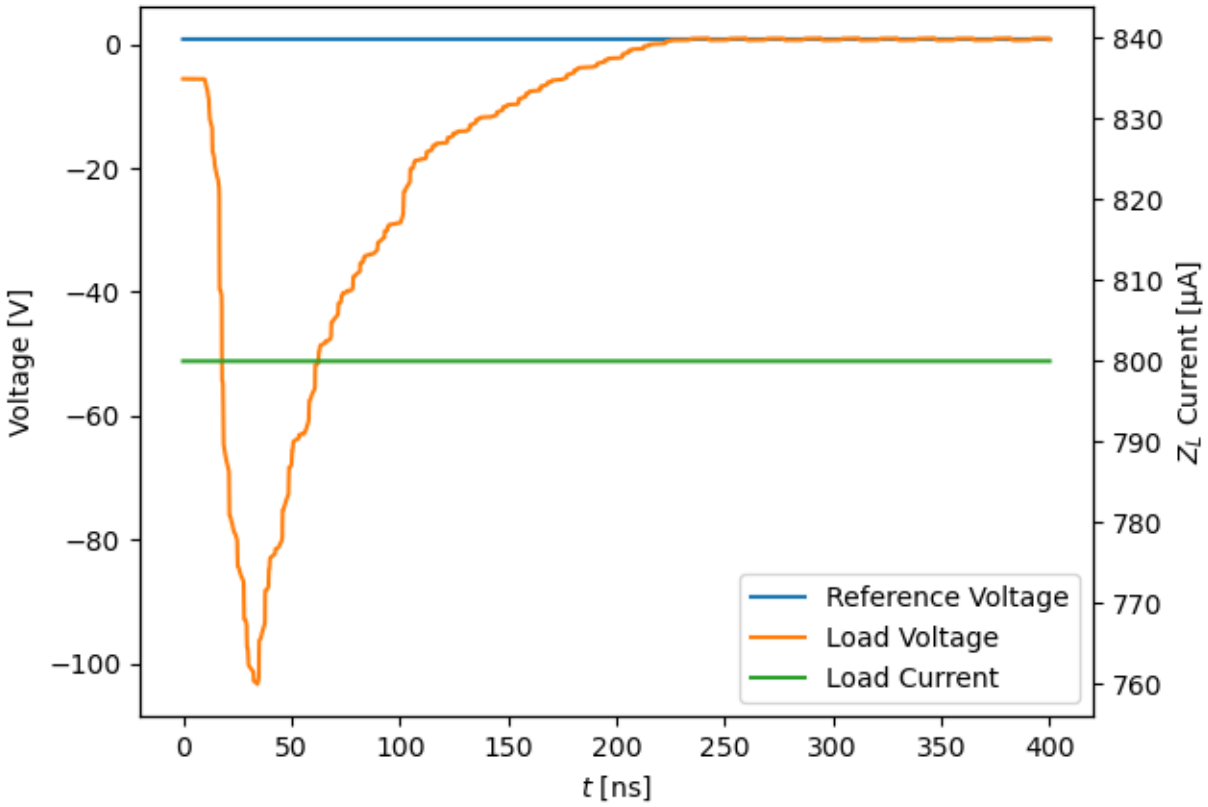

Figure 2.10: Transient response of integral control digital LDO.

The large drop in voltage at the start of the testbench is an artifact of the test, since all pFETs are set to open at the start of the testbench and it is unlikely that there will be instant current load at circuit startup. The settling time of the LDO is approximately 200 ns. Using Equation 2.3, the theoretical settling time should be 160 ns, making the design perform 25 % worse than predicted. This is likely due to the clock-to-Q time of the comparator and the finite rise and fall time of the clock. Restricting the plot to values beyond $t = 200$ ns, we can observe the limit cycle.

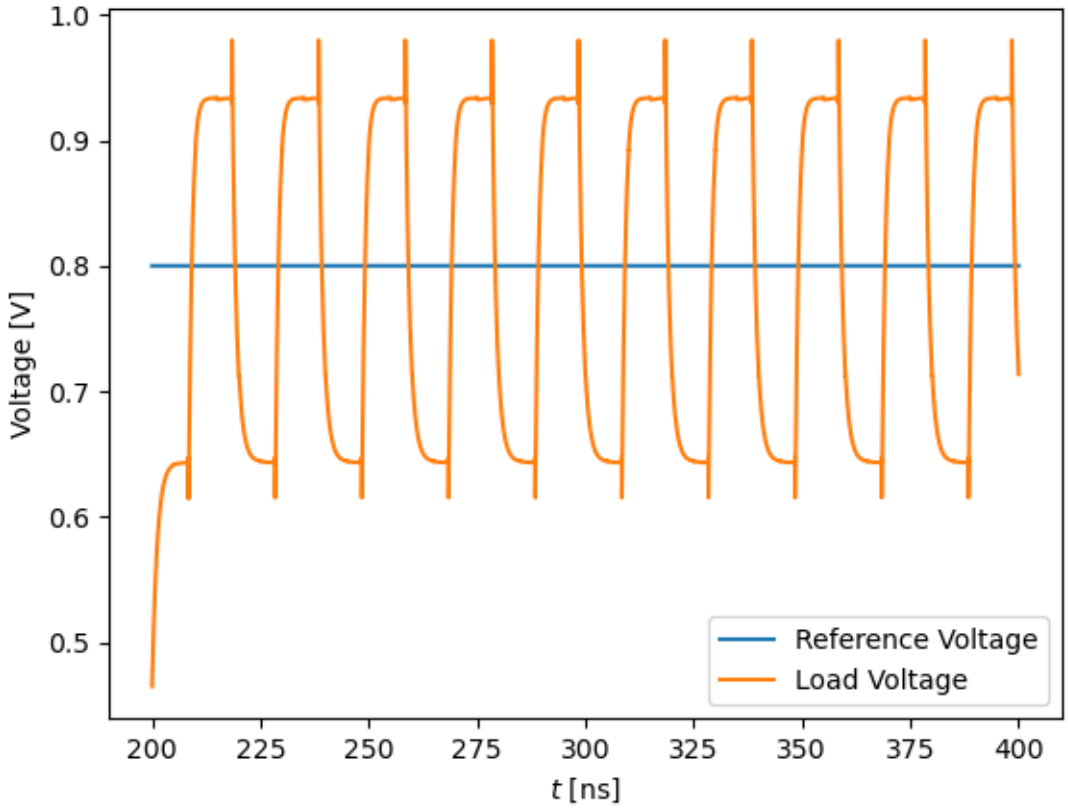

Figure 2.11: Transient response limit cycle.

As predicted, the digital LDO has a limit cycle centered around the desired reference voltage with a peak-to-peak variance of approximately 300 mV. Changing the reference voltage to 1 V, we observe that the limit cycle is 200 mV, suggesting that the relationship is roughly constant.

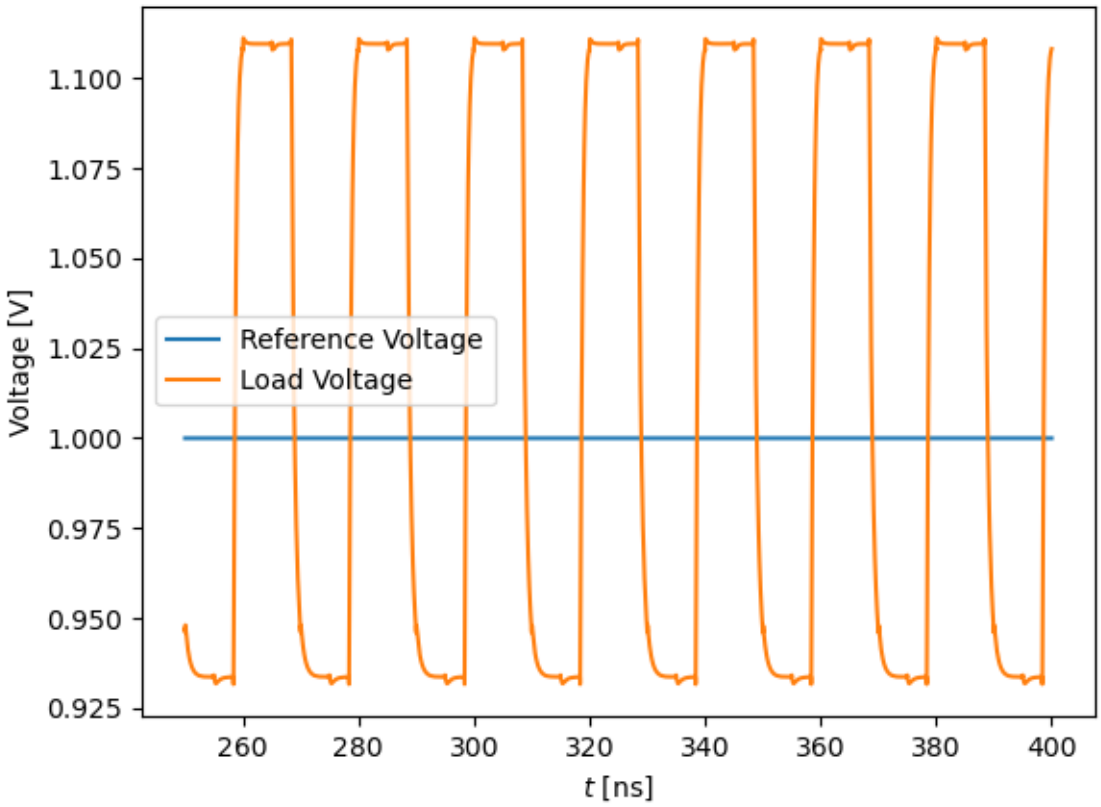

Figure 2.12: Transient response limit cycle with $V_{\text{ref}} = 1\,\text{V}$.

The dropout voltage of the design can be measured by setting the input voltage to the supply voltage and letting the LDO settle to a final value. The difference between the supply and this load voltage is the dropout voltage. As the name suggests, a key design parameter is minimizing this value, as it means implementation of this circuit allows a wide operating range with little overhead.

	[7]	[10]	[13]	[14]	**This Work**
Process	65 nm	65 nm	65 nm	65 nm	130 nm
Control Type	Linear	AGT	SAR/PWM/PD	Linear	Linear
Clock Frequency [MHz]	1–10	16–100	1–240	200	100
Settling Time [µs]	240	0.028	0.1	1.1	0.2
Dropout Voltage [mV]	50	50	200	50	144
Max Current [mA]	0.2	16.3	2	15	2.8
C_L [nF]	100	0.25	0.4	N/A	1×10^{-4}
Quiescent Current [µA]	2.7	80–1200	14	N/A	10
Current Efficiency [%]	98.7	N/A	99.8	96.4	99.6

Table 2.1: Comparison of this work against related works.

2.4 Integration

With some work, it would be possible to integrate this digital LDO into Hammer's flow, specifically the multi-domain flow described in the previous chapter. Since the control logic has already been placed and routed using Hammer, it can simply be dropped in as a sub-block within a larger hierarchy on chip. The comparator layout and pFET can be generated with a tool such as Berkeley Analog Generator (BAG) [15] and integrated similarly. They can then be abstracted and connected to each other as routes.

As for specifying their domains, the supply voltage of the LDO itself would be connected to one domain, likely the always-on domain. Then, the load voltage that the LDO supplies would exist as a second domain, depending on the application. This can be imagined as an active level shifter that can track across a wider range of current loads.

Chapter 3

Conclusion

The first part of this work demonstrated the modifications required to an open-source VLSI flow tool to partially support multi-power domain designs. Primarily, new power format generators and configuration file APIs are required for the user to specify a now more complex power grid in physical design. Furthermore, special cells must be added and given to the PAR tool to be able to generate proper nets. We show that an industry-standard tool, such as Cadence Innovus, is able to ingest a power intent generated by the configuration file and recognize these domains.

The second part of this work demonstrates the design and implementation of a digitally-controlled LDO in 130 nm CMOS to be used in a low-power design. The design targets medium-frequency low-power applications with a quiescent current of 10 µA and a settling time of 200 ns. When using integral control law logic, the design functions at a wide range of voltage inputs and currents. The circuit topology is highly modular with the ability to scale key parameters and figures of merit. We also demonstrate the feasibility of open-source process technologies for mixed-signal design.

3.1 Future Work

The first part of this work demonstrates the steps for implementing multi-power domain design in the open-source VLSI tool Hammer. However, work needs to be done integrating the process technology into this new flow. Once the relevant standard cells for low-power VLSI, such as level shifters and isolation cells are declared, the relevant scripting commands for a PAR tool should be able to place down these cells automatically to generate truly distinct nets.

With regards to this digital LDO design in the second part of this work, there are several improvements that could be made. The SAR algorithm would provide a substantial improvement in settling time. Furthermore, brute-force improvements can be made by adding more pFETs into the array to improve maximum current. Current efficiency can be improved by using digitally-based comparator logic instead of a mixed-signal design, such as in [14], since

there would no longer be direct paths from supply to ground during a clock cycle. The error granularity can be increased by scaling the comparator up to a multi-bit ADC, of which there are many designs to choose from to optimize clock-to-Q, quiescent current, and further parameters. Finally, there is work to be done to get the design to a state that can be taped out and measured on a physical die.

Bibliography

[1] [Online]. Available: https://skywater-pdk.readthedocs.io/en/main/.

[2] H. Liew, D. Grubb, J. Wright, *et al.*, "Hammer: A modular and reusable physical design flow tool: Invited," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, ser. DAC '22, San Francisco, California: Association for Computing Machinery, 2022, pp. 1335–1338, ISBN: 9781450391429. DOI: 10.1145/3489517.3530672. [Online]. Available: https://doi.org/10.1145/3489517.3530672.

[3] "Ieee standard for design and verification of low-power, energy-aware electronic systems," *IEEE Std 1801-2018*, pp. 1–548, 2019. DOI: 10.1109/IEEESTD.2019.8686430.

[4] L. F. W. Group, Dec. 2014. [Online]. Available: http://www.si2.org/wp-content/uploads/2020/04/si2_cpf_v2.1_29-dec-2014.pdf.

[5] [Online]. Available: https://github.com/ucb-bar/hammer/tree/bryan-multi-power.

[6] A. Raychowdhury, "T2: Fundamentals of digital and digitally assisted linear voltage regulators," 2024.

[7] Y. Okuma, K. Ishida, Y. Ryu, *et al.*, "0.5-v input digital ldo with 98.7% current efficiency and 2.7-µa quiescent current in 65nm cmos," in *IEEE Custom Integrated Circuits Conference 2010*, 2010, pp. 1–4. DOI: 10.1109/CICC.2010.5617586.

[8] B. Razavi, "The strongarm latch [a circuit for all seasons]," *IEEE Solid-State Circuits Magazine*, vol. 7, no. 2, pp. 12–17, 2015. DOI: 10.1109/MSSC.2015.2418155.

[9] Y. Lee, A. Waterman, H. Cook, *et al.*, "An agile approach to building risc-v microprocessors," *IEEE Micro*, vol. 36, no. 2, pp. 8–20, 2016. DOI: 10.1109/MM.2016.11.

[10] L. G. Salem, J. Warchall, and P. P. Mercier, "A successive approximation recursive digital low-dropout voltage regulator with pd compensation and sub-lsb duty control," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 1, pp. 35–49, 2018. DOI: 10.1109/JSSC.2017.2766215.

[11] Z. Wang, S. J. Kim, K. Bowman, and M. Seok, "Review, survey, and benchmark of recent digital ldo voltage regulators," in *2022 IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 01–08. DOI: 10.1109/CICC53496.2022.9772734.

[12] C. Sodini, P.-K. Ko, and J. Moll, "The effect of high fields on mos device and circuit performance," *IEEE Transactions on Electron Devices*, vol. 31, no. 10, pp. 1386–1393, 1984. DOI: 10.1109/T-ED.1984.21721.

[13] X. Sun, A. Boora, W. Zhang, V. R. Pamula, and V. Sathe, "14.5 a 0.6-to-1.1v computationally regulated digital ldo with 2.79-cycle mean settling time and autonomous runtime gain tracking in 65nm cmos," in *2019 IEEE International Solid-State Circuits Conference - (ISSCC)*, 2019, pp. 230–232. DOI: 10.1109/ISSCC.2019.8662298.

[14] Y. K. Cherivirala, M. Saligane, and D. D. Wentzloff, "An open source compatible framework to fully autonomous digital ldo generation," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2023, pp. 1–5. DOI: 10.1109/ISCAS46773.2023.10181884.

[15] E. Chang, J. Han, W. Bae, *et al.*, "Bag2: A process-portable framework for generator-based ams circuit design," in *2018 IEEE Custom Integrated Circuits Conference (CICC)*, 2018, pp. 1–8. DOI: 10.1109/CICC.2018.8357061.