Mini Project 3

7-Bit MOSFET Based Digital to Analog Converter

Ari Bobesh, Ahan Trivedi, Ertug Umsur

Source Files: https://github.com/Ahan-Trivedi2/mosfet-dac

I. Introduction:

In this project, we designed a seven-bit current-output digital-to-analog converter (DAC) using a MOSFET-based R–2R-like ladder network implemented in the SkyWater 130 nm CMOS (sky130pdk) process. The DAC operates from a 1.8 V single-ended supply and converts seven ideal digital input bits into an analog output current that can source or sink with less than 0.5%/V variation over at least 80% of the supply range and less than 0.5%/V dependence on supply voltage. The design targets differential and integral nonlinearities (DNL/INL) below 2 LSB and 4 LSB, respectively, under nominal conditions, and within 8 LSB under mismatch as verified through Monte Carlo simulations in Ngspice. A single external resistor provides the bias current, and the design passes both DRC and LVS checks. The layout follows analog matching best practices—such as symmetry, dummy devices, and selective common-centroiding—to minimize systematic error. The architecture is based on a MOSFET-only implementation of the traditional R–2R ladder, inspired by Hammerschmied and Huang's untrimmed 10-bit DAC.

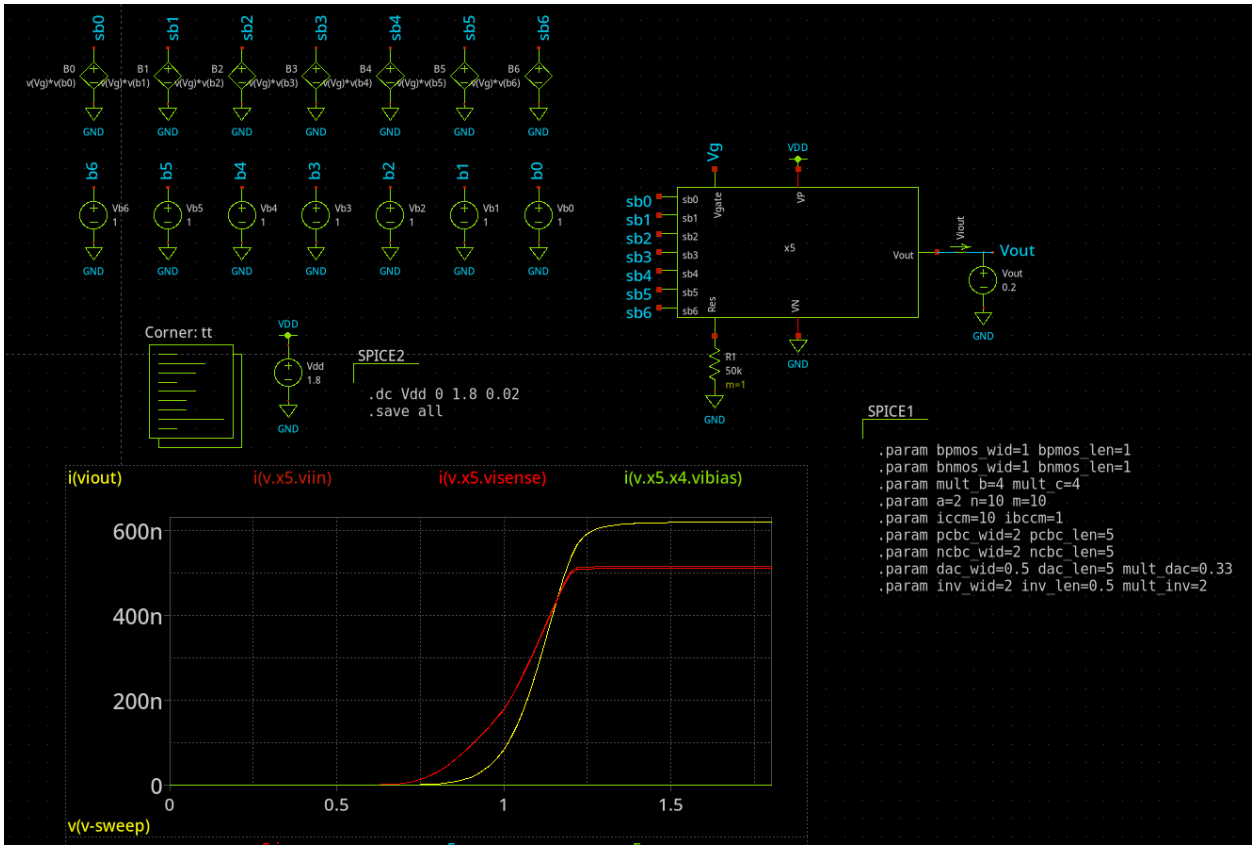
II. Schematic Capture and Simulation

This project was built around a hierarchical structure in both schematic and layout. For simplicity of understanding, here is a breakdown of the project structure:

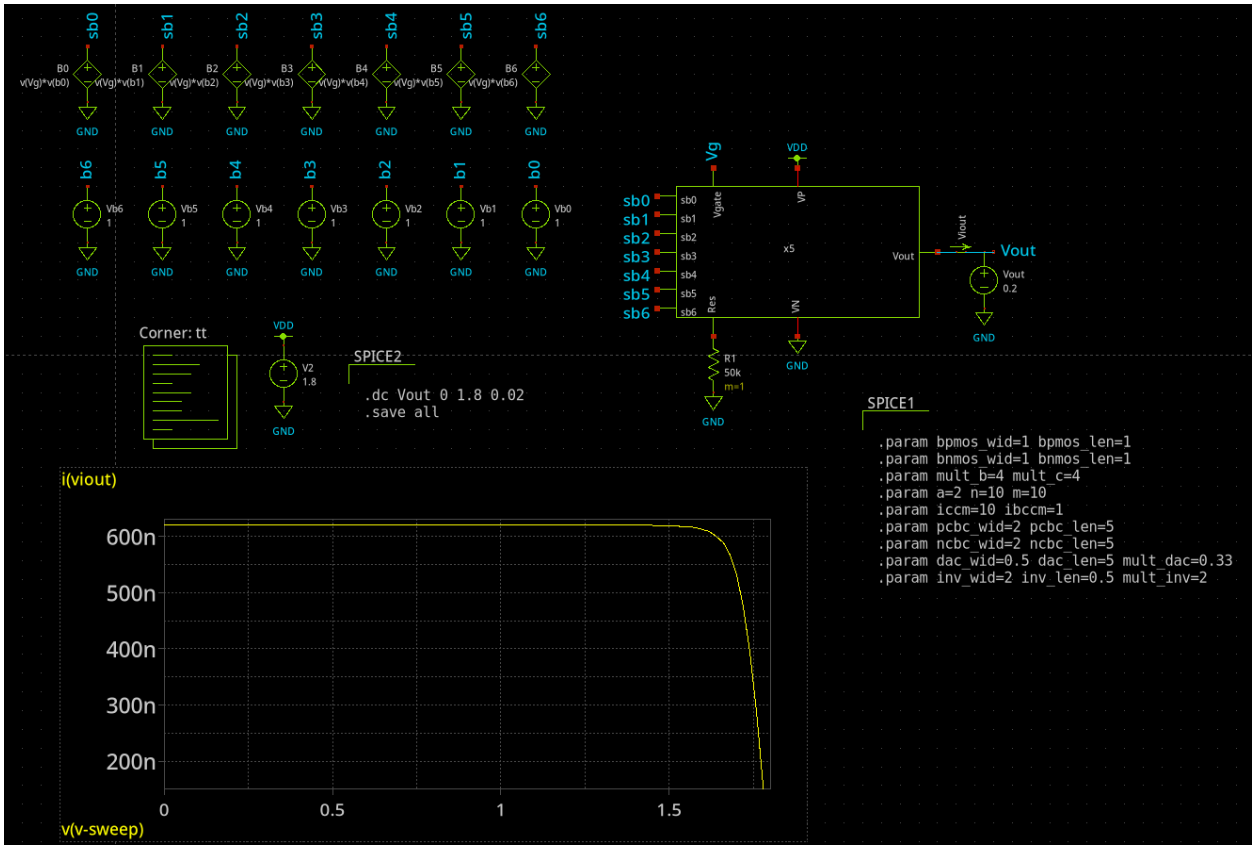
- **LDS:** cleaned schematics for LDS
- **Layout:** MAGIC VLSI layouts for the LDS
- **.:** all the schematics and testbench files

During the creation of the DAC, testbenches were used to categorize the different components individually and then altogether. Three main testbench styles were used to isolate the different testing characteristics.

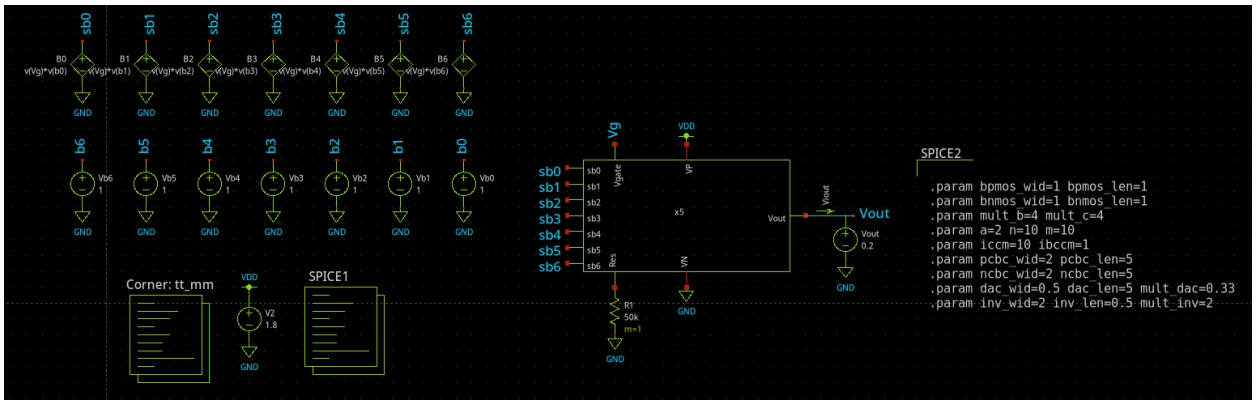
The VDD sweep test bench sweeps the VDD source while measuring the output of the subcircuit. This checks to ensure that the output is supply independent. The image below shows the testbench in action. The final DAC manages to have a stable output at around 0.7%/V for more than 20% of the VDD range, which is a tradeoff with linearity since the FVF characteristics and the inversion level of the transistors affect the VDD dependence strongly.



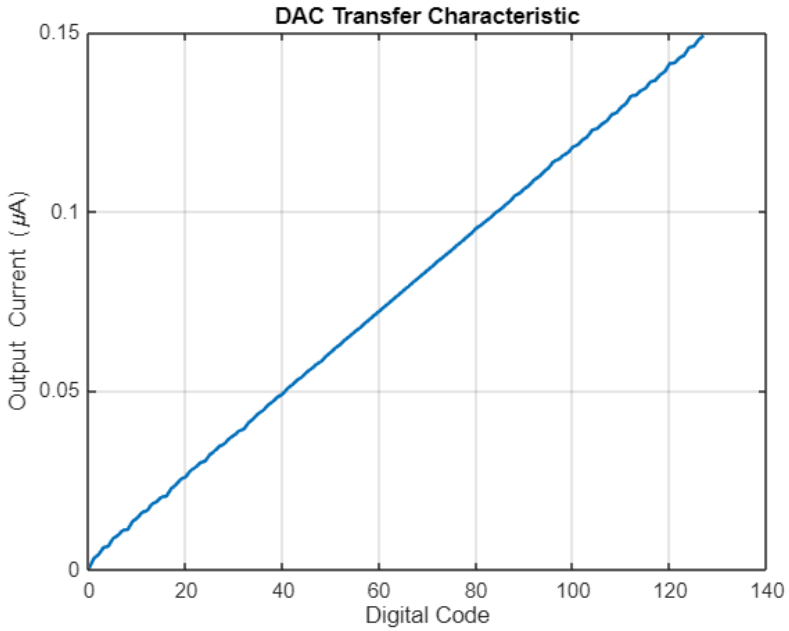
The Vout sweep test bench sweeps the Vout source while measuring the output of the subcircuit. This checks to ensure that the output is unaffected. The image below shows the testbench in action. The final DAC manages to have a stable output at around 0.03%/V for more than 80% of the Vout range. This flatness is insanely good and was not the limiting factor in the design as we progressed.



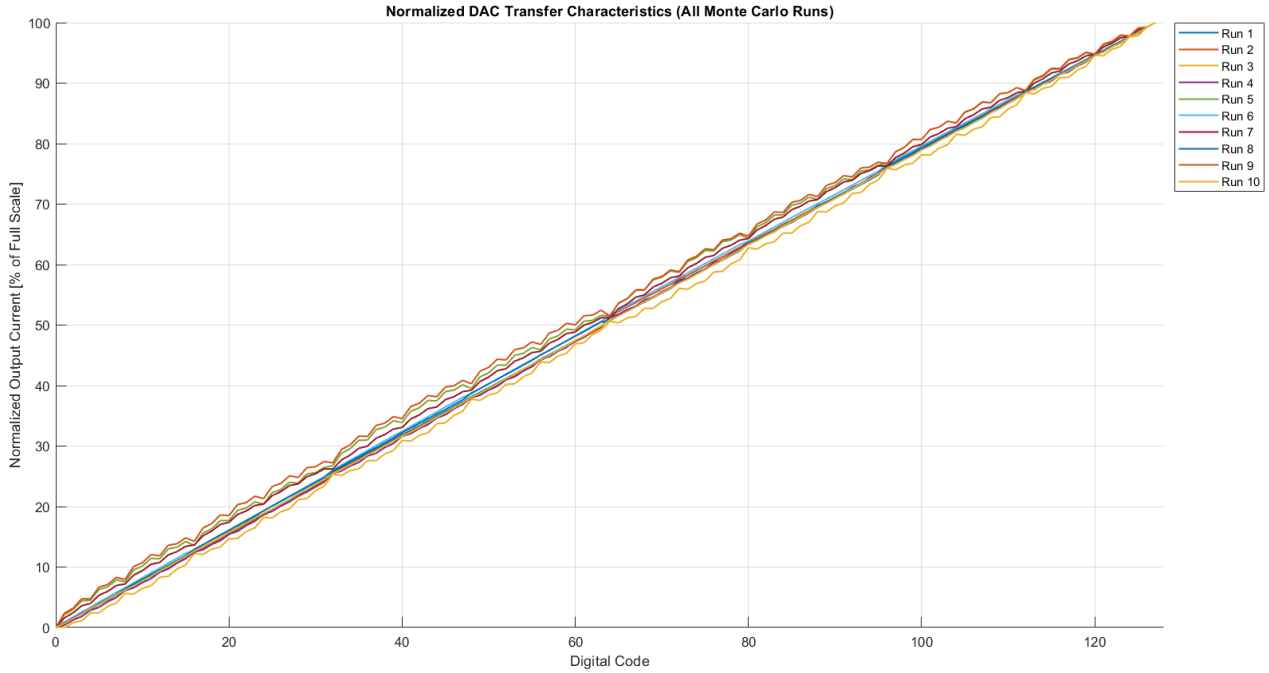
The linearity sweep test bench sweeps the bits while measuring the output of the subcircuit. This checks to ensure that the output is linear. The image below shows the testbench in action. The final DAC manages to have a linear output with a worst-case DNL of 2.23 LSB and a worst-case INL of 5.72 LSB with mismatched models. To achieve this, we had to parallelize a lot of transistors and drastically increase the current through the DAC from nA to uA.



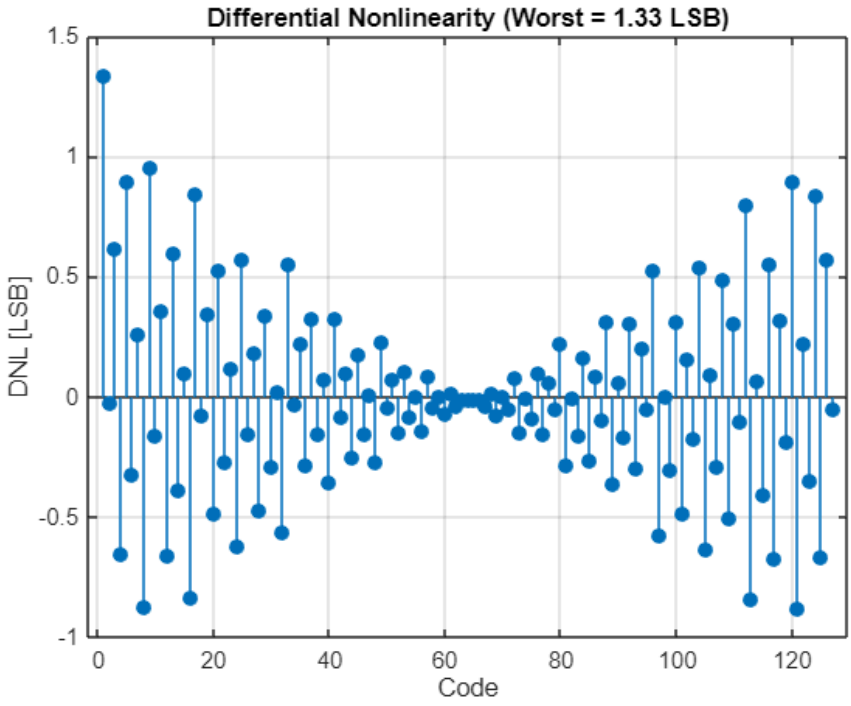
Below are some example graphs from our linearity testing.



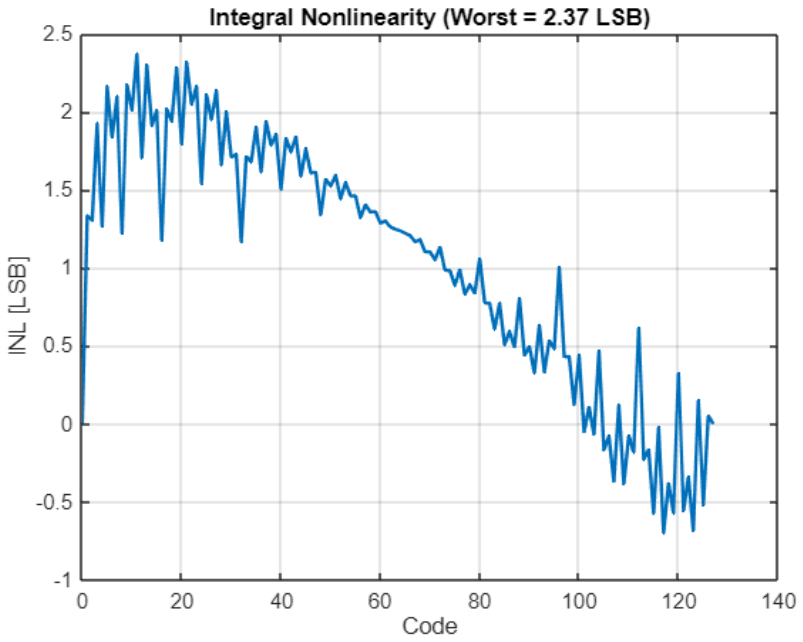
Nominal DAC Transfer Characteristic showing output current as a function of digital code showing linear behavior.



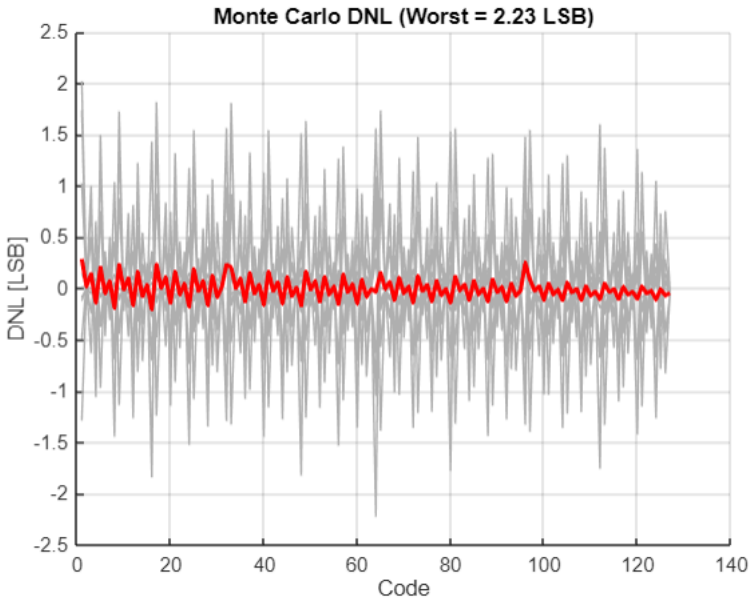
Normalized output current versus digital input code for 10 Monte Carlo DAC simulations. Each run is scaled to its full-scale range (0–100%) to highlight variations in linearity and code transition uniformity across process mismatch.



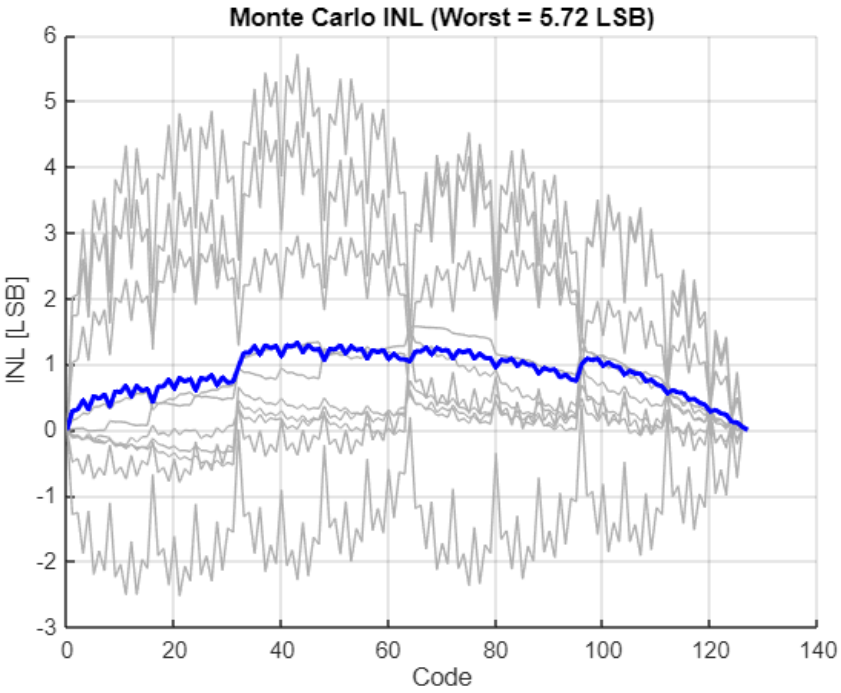
Differential Nonlinearity (DNL) of the nominal DAC output. Each point represents the deviation in step size between adjacent codes relative to one ideal LSB. The nominal design exhibits a worst-case DNL of 1.33 LSB, meeting project specifications.



Integral Nonlinearity (INL) of the nominal DAC output. Each point represents the cumulative deviation of the actual transfer function from the ideal linear response, measured in LSBs. The nominal design exhibits a worst-case INL of 2.37 LSB, meeting project specifications.



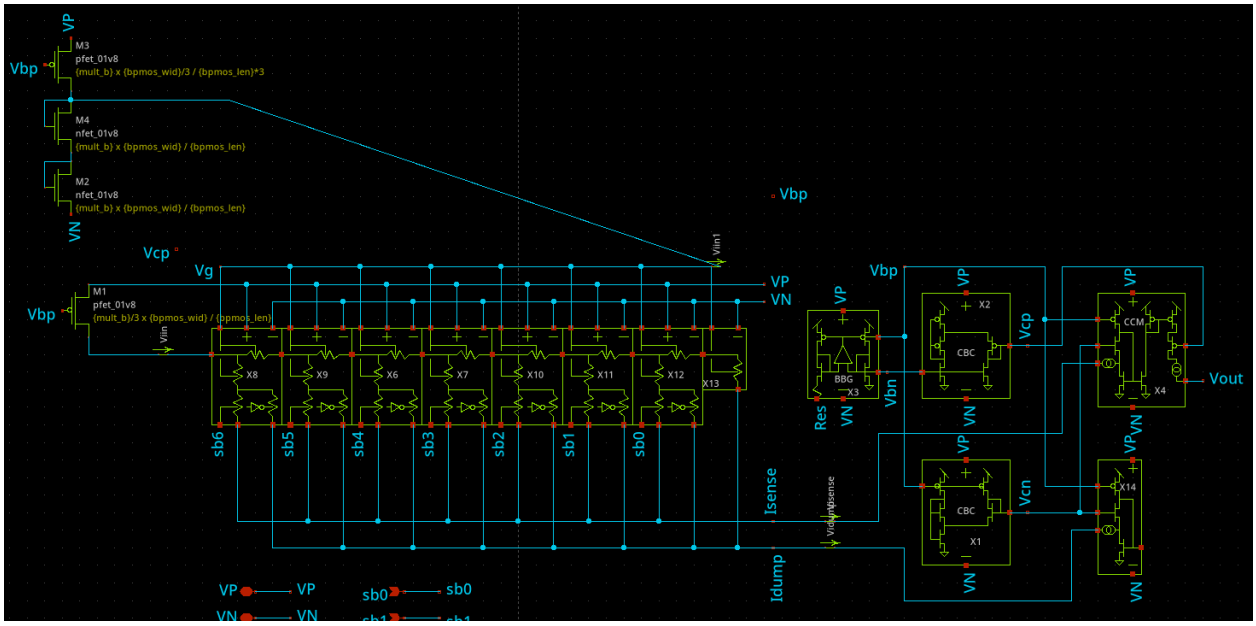
Monte Carlo Differential Nonlinearity (DNL) of the DAC. Each gray trace represents DNL variation from an individual Monte Carlo simulation run, while the red line shows the mean DNL across all runs. The worst-case DNL observed is 2.23 LSB, demonstrating increased step-size variation due to device mismatch effects.



Monte Carlo Integral Nonlinearity (INL) of the DAC. Each gray curve represents the INL profile from an individual Monte Carlo simulation run, while the blue line indicates the mean INL across all runs. The worst-case INL of 5.72 LSB reflects cumulative deviation from ideal linearity under process mismatch conditions.

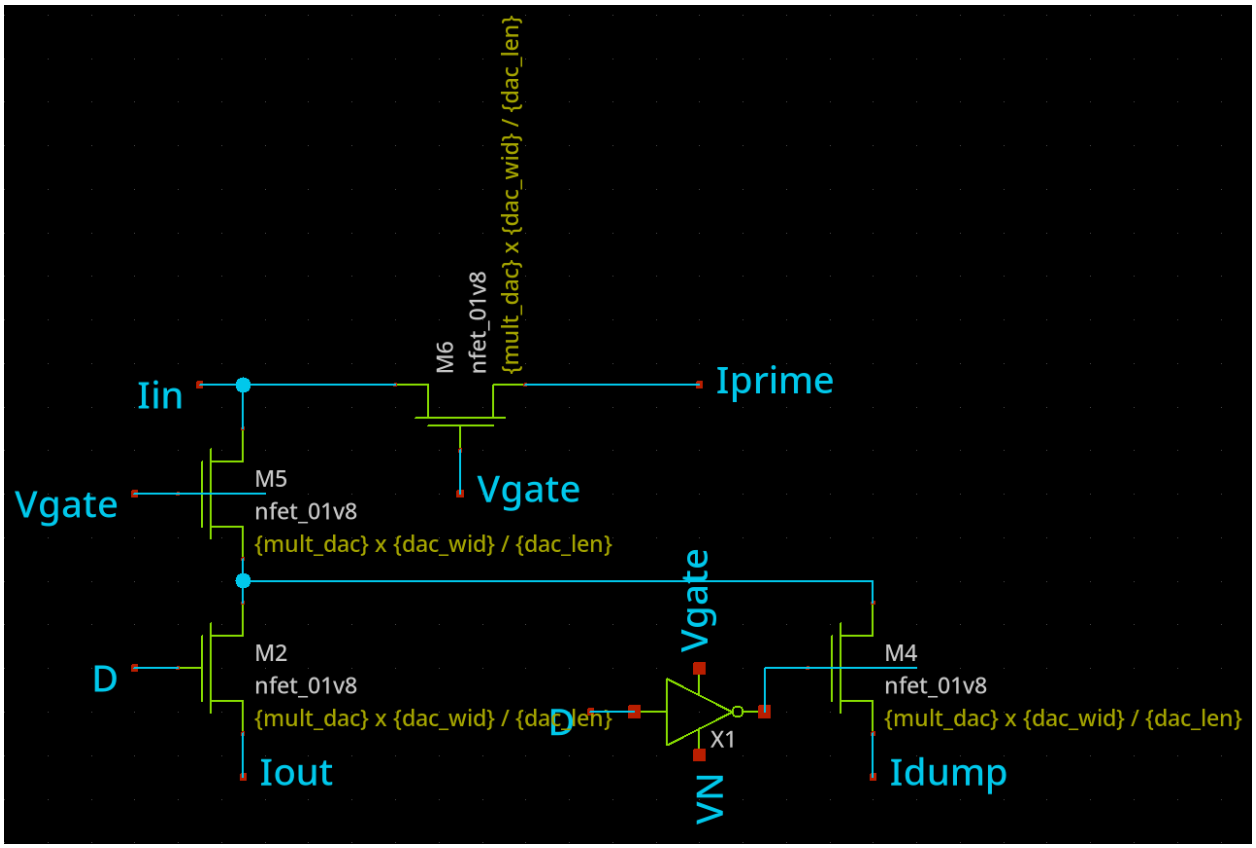
The DAC circuit contains seven bits and a few notable subcircuits including:

- Bootstrap bias generator (BBG)
- Positive cascode bias circuit (PCBC)
- Negative bootstrap bias circuit (NCBC)
- Flipped voltage follower (FVF)
- Cascoded current mirror (CCM)
- DAC ladder block/end
- Miscellaneous biasing at the top level

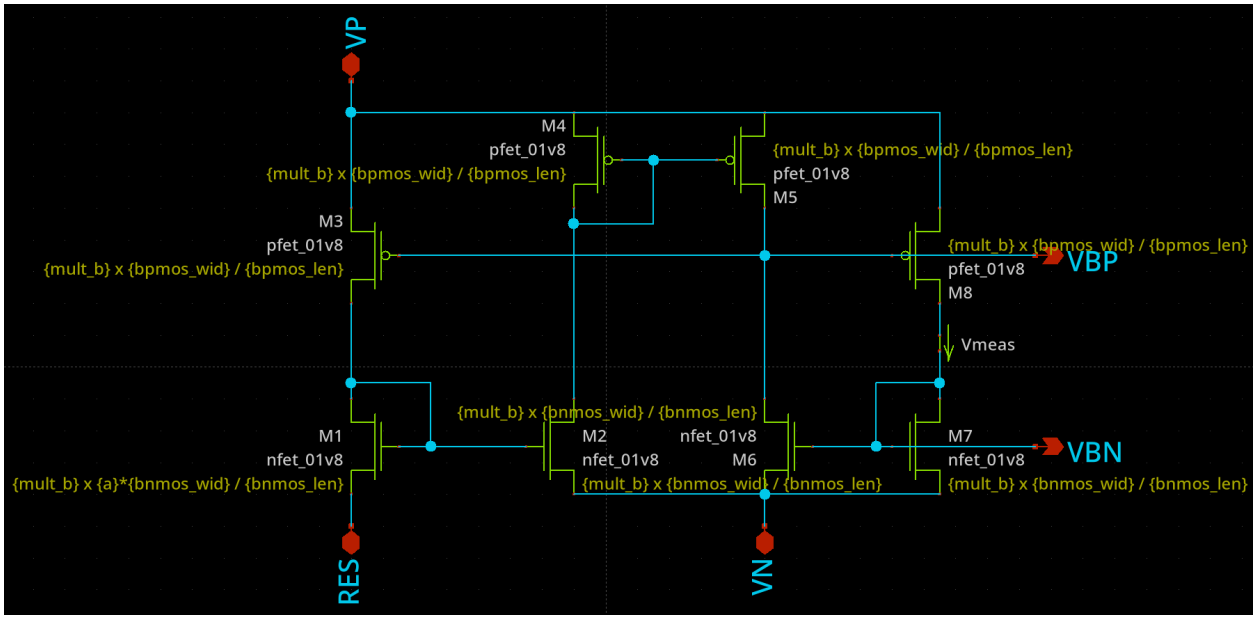


The greatest improvement to DAC linearity in our experimentation came from holding the DAC network transistors at a weaker inversion level than full VDD - reducing the impact of the FVF voltage mismatch between the sense and dump lines. This was attempted through various biasing methods, but it was found that a constant voltage was not the best way to achieve this, especially within the context of the VDD sweep. We noticed that we could use a doubly stacked diode connected MOSFET ladder to get the voltage we needed and the characteristic we were looking for.

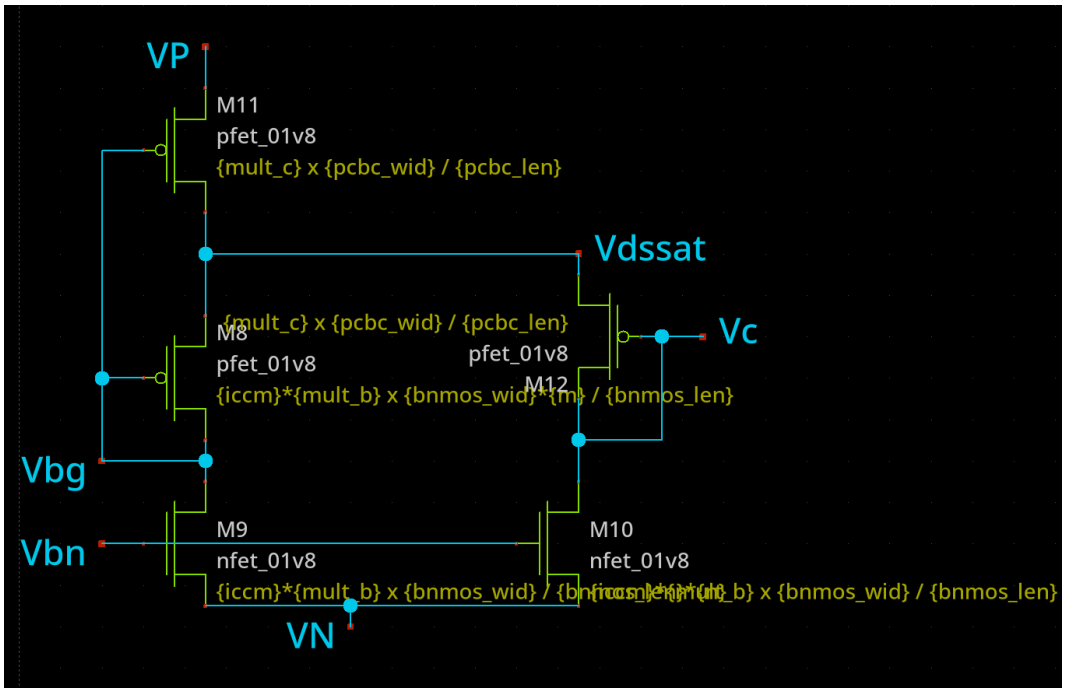
The DAC ladder consists of a R-2R ladder network made entirely of NMOS transistors. An inverter allows each bit to have complementary signaling to switch between the sense and dump lines. Importantly, the inverter must be powered off the gate voltage. The end block, not pictured, ends the R-2R network with the proper termination to the dump line. This design is very unique because it uses the pass transistors as part of the R-2R network. This allows you to theoretically kill two birds with one stone and get superior matching across the network.

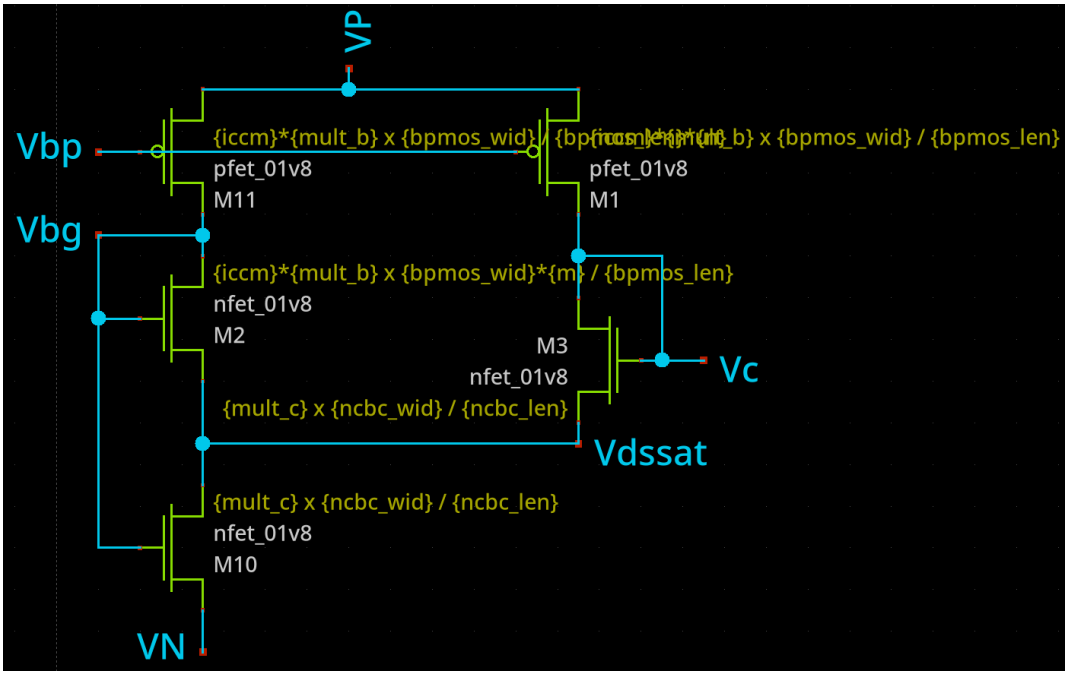


The BBG generates a constant bias current regardless of the supply voltage. It works by strapping an amplifier on a degenerated NMOS mirror, giving it extremely consistent performance. The circuit is really sensitive to changes in transistor width, so the only real way to make it more reliable is parallelization to avoid messing with the threshold voltages. The currents are mirrored into other parts of the circuit via transistor multiplicity to ensure that they are appropriately matched multiples as needed. This subcircuit generates both the positive and negative bias currents.

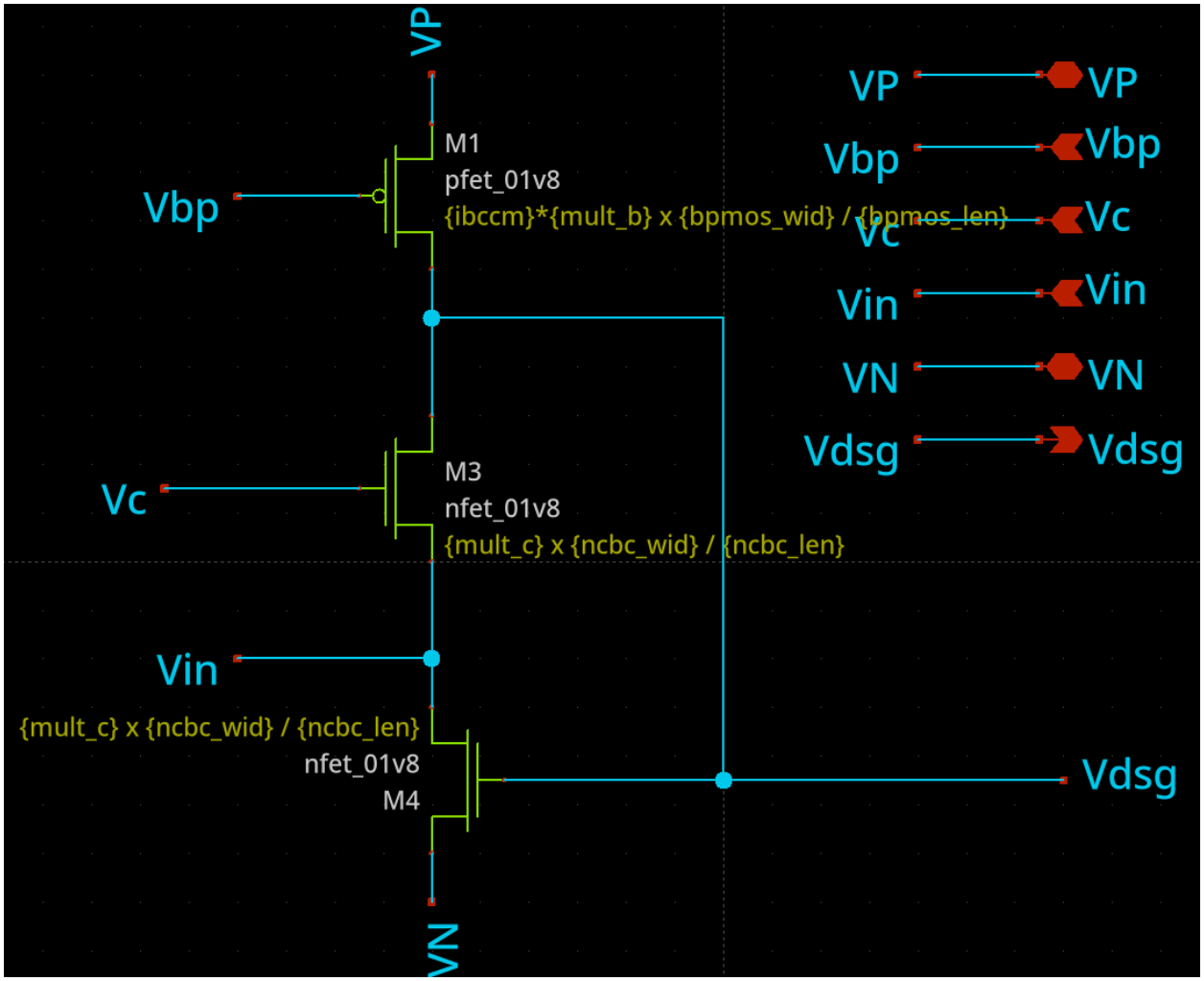


The cascode biasing circuits are the main way we get output voltage independence across a large voltage range. They maintain a cascode at the appropriate voltage to be right at V_{DSSAT}. These two complementary circuits generate both the positive and negative biases. It is important that they are primed with a larger current than is expected through the cascode current mirror during normal operation. This slightly decreases headroom and makes the non-linearity start sooner, but ensures that the cascode will never end up stifled.

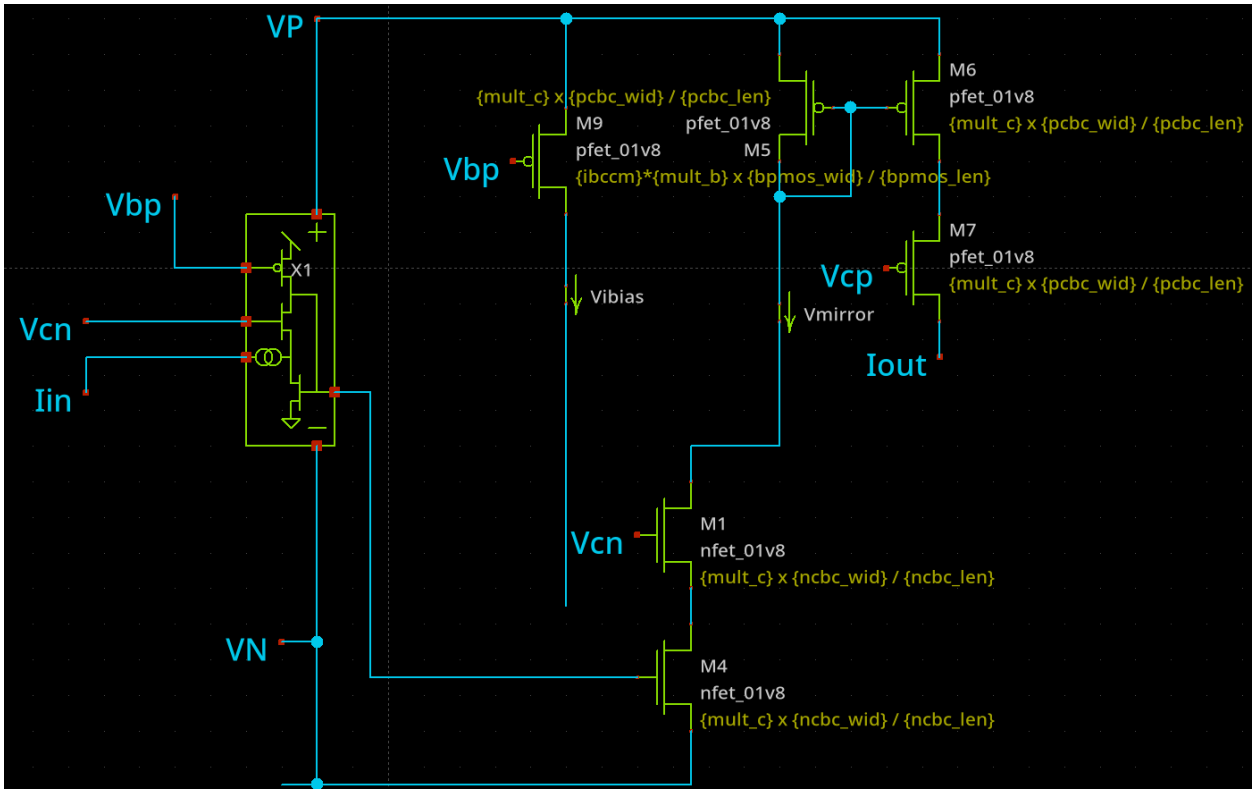




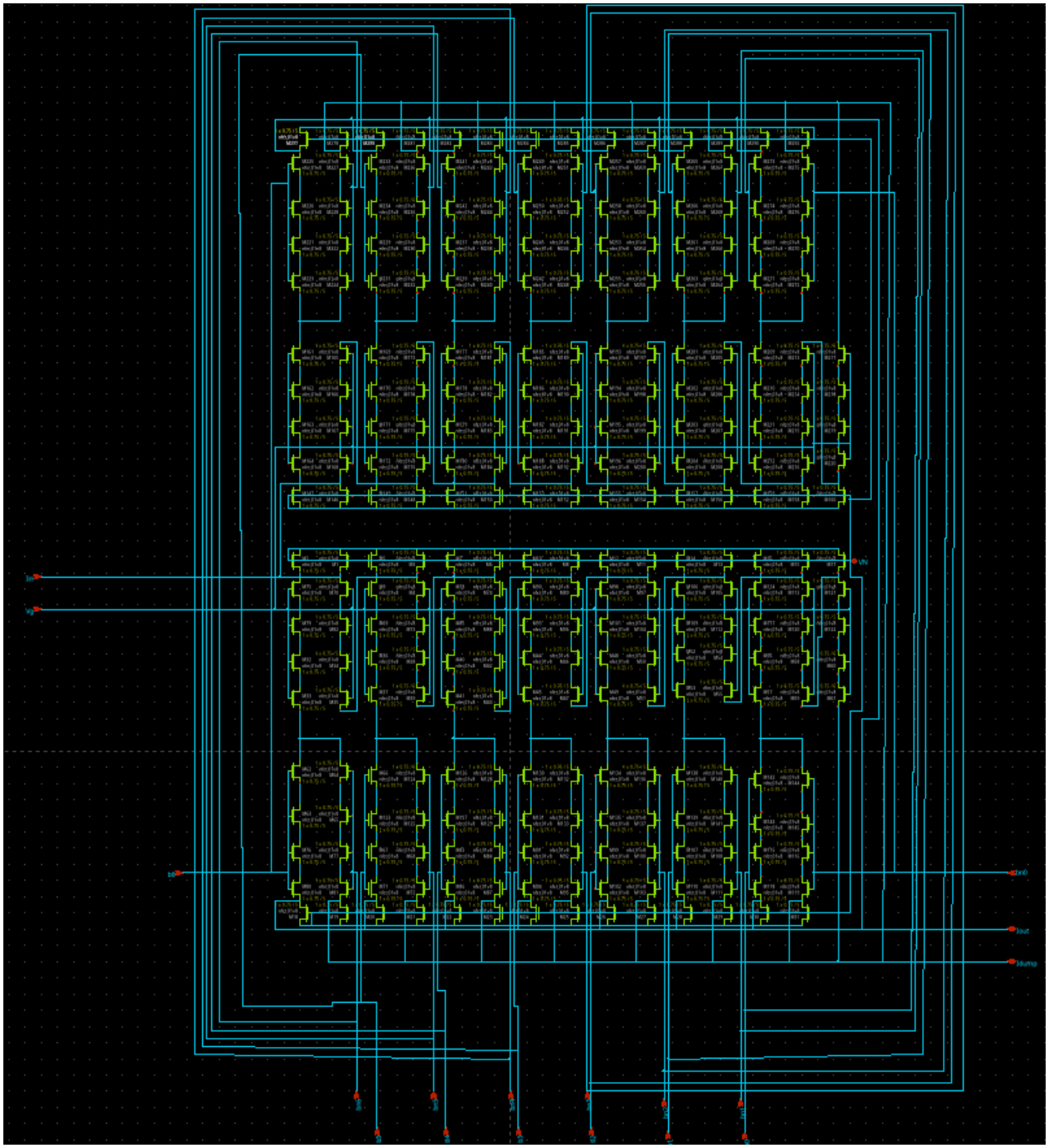
The FVF is the main crux of this topology. It is essential to maintain the same voltage on the sense and dump lines regardless of their currents. Otherwise, the DAC ladder experiences major distortions from the source transconductance of the FETs. The FVF provides a really low input impedance, but also injects a non-negligible bias current into the output. It is best to keep this bias current as low as possible to reduce its effects, but high enough such that it can keep the FVF active in the expected operating regions - too low and the input impedance rises. While very good, the FVF is up against the source transconductance, so literally any mV of difference means the difference between sun and moon for the linearity of the DAC as a whole.



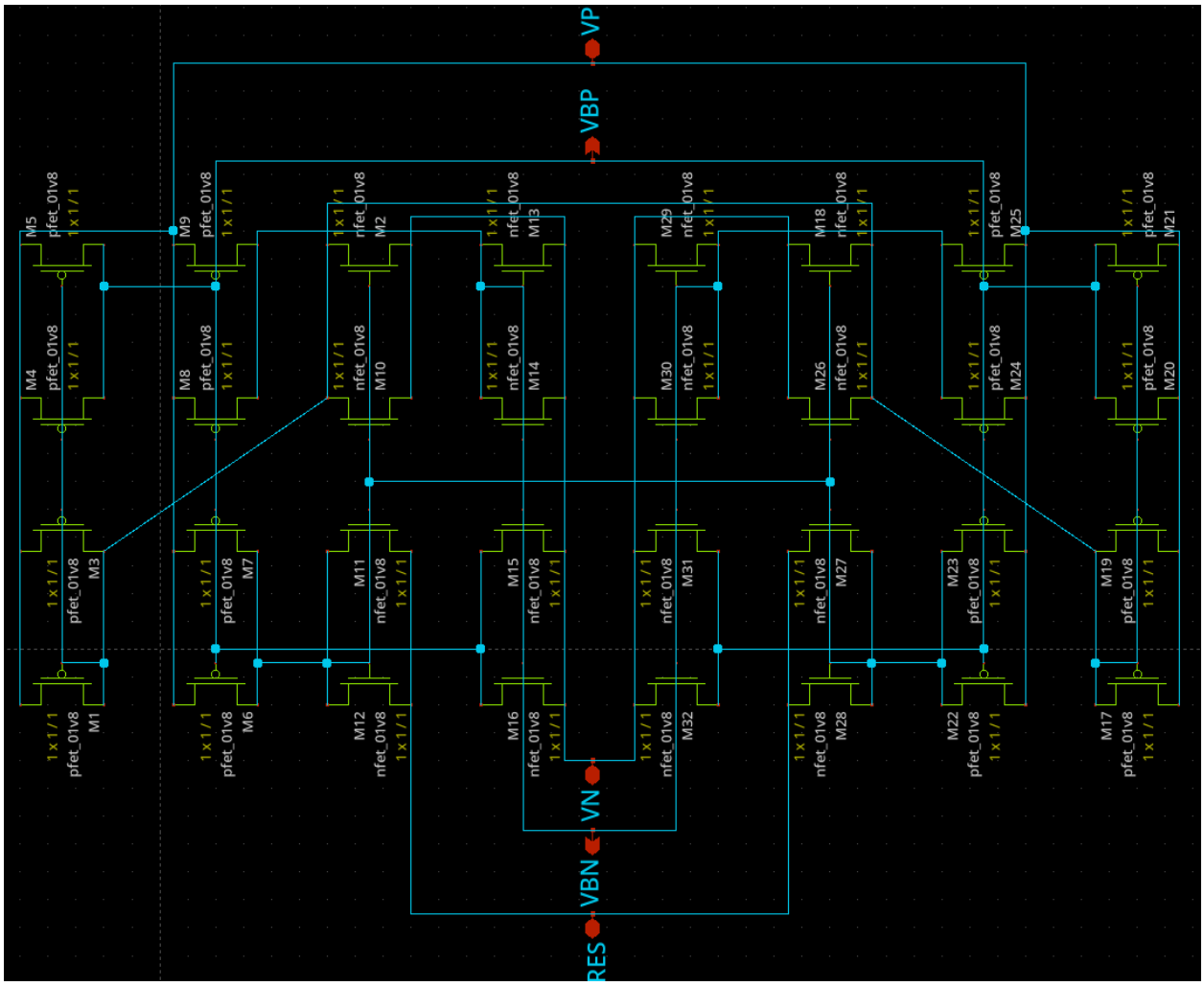
The cascode current mirror employs the FVF input and two cascodes to get a good output characteristic. It was found that the NMOS transistor models did not operate well in a cascode, so the output was mirrored again with a current mirror into a PMOS cascode instead. The NMOS cascode was kept because removing it made the NMOS mirror behave weirdly when faced with VDD changes - the early effect was too great from the PMOS mirror. The bias subtractor was removed after it was shown that despite the best matching efforts the FVF bias could not be accurately subtracted without resulting in negative currents and a corresponding zero-output region.



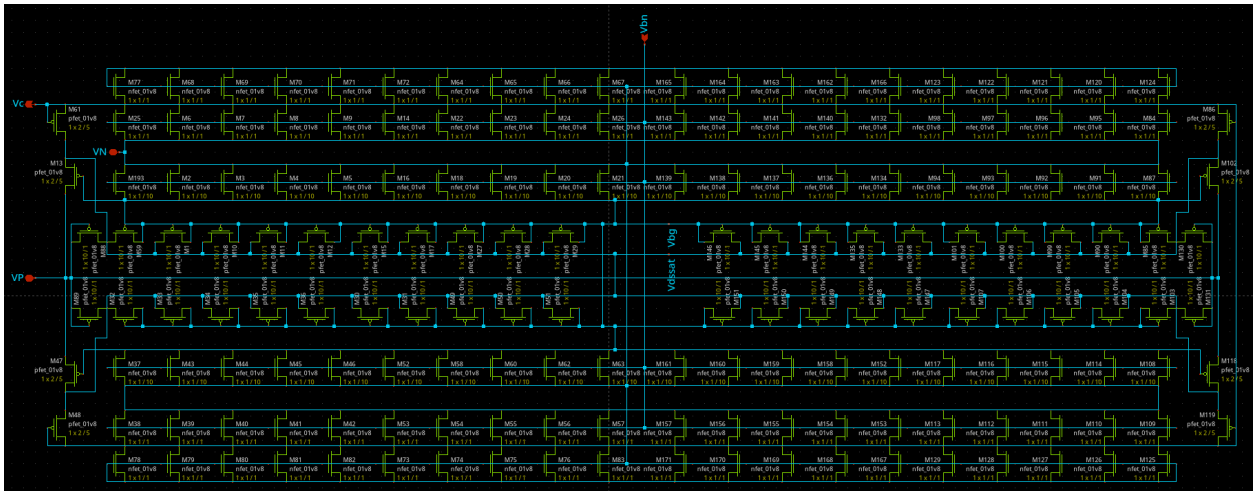
Before moving into layout we transferred all the prior modules into layout driven schematics, a few examples pictured below. These were made using dummy transistors and common centroid techniques, helping us organize our layouts.



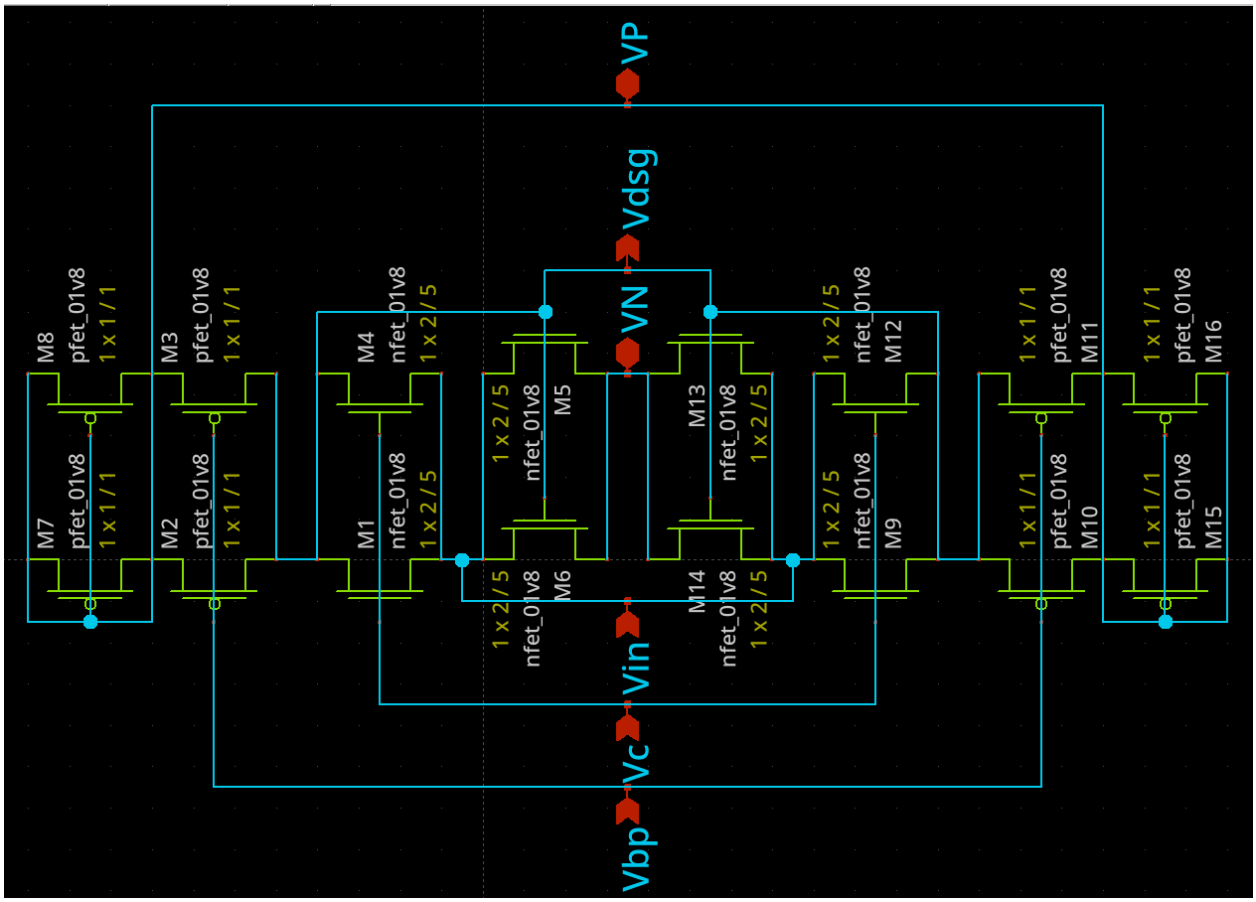
The DAC



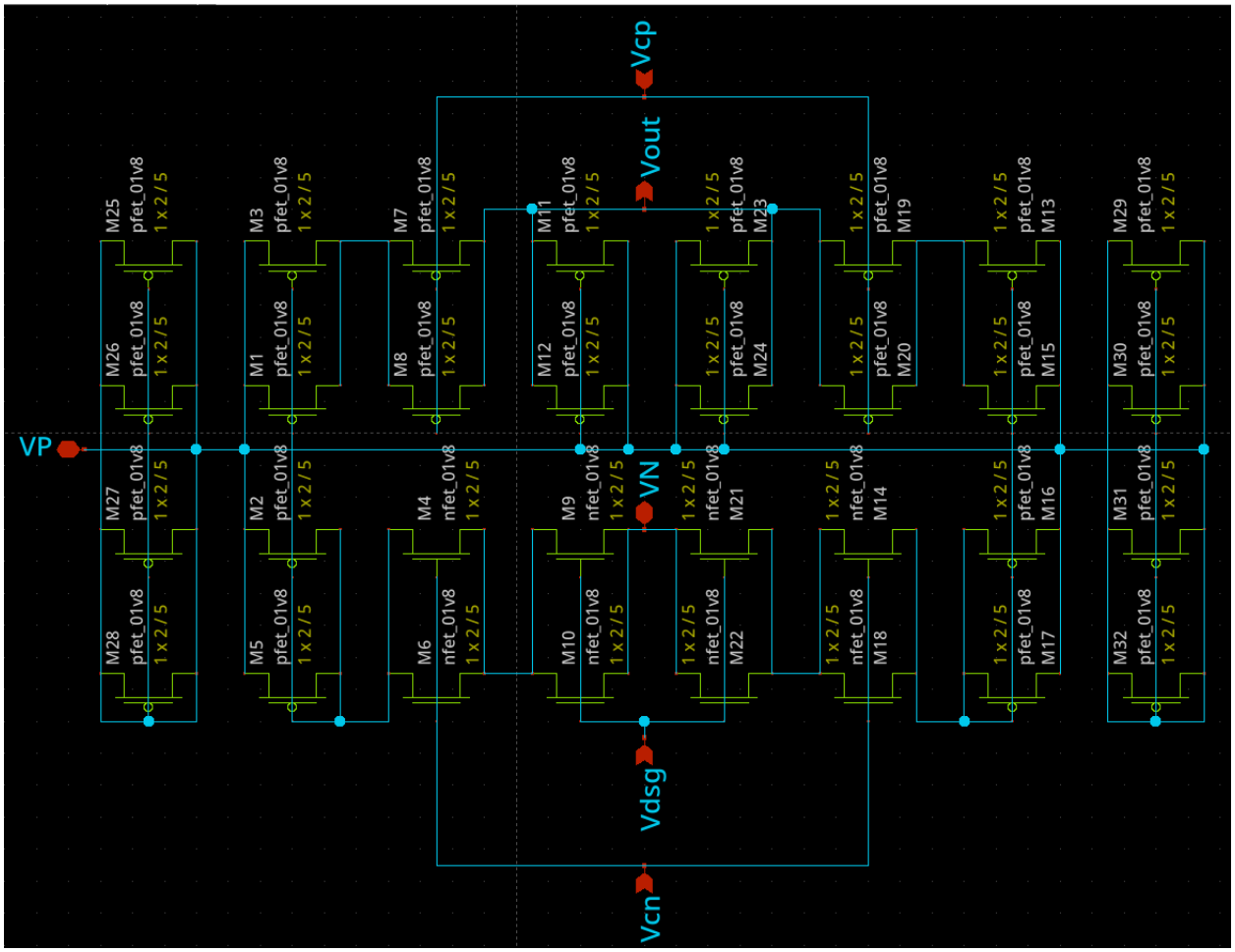
The BBG



The PCBC


VP
Vdsg
VN
Vin
Vc
Vbp
M8
pfet_01v8
1 x 1 / 1
M3
pfet_01v8
1 x 1 / 1
M4
nfet_01v8
1 x 2 / 5
M12
nfet_01v8
1 x 2 / 5
M11
pfet_01v8
1 x 1 / 1
M16
pfet_01v8
1 x 1 / 1
M7
pfet_01v8
1 x 1 / 1
M2
pfet_01v8
1 x 1 / 1
M1
nfet_01v8
1 x 2 / 5
M5
nfet_01v8
1 x 2 / 5
M13
nfet_01v8
1 x 2 / 5
M9
nfet_01v8
1 x 2 / 5
M10
pfet_01v8
1 x 1 / 1
M15
pfet_01v8
1 x 1 / 1
M6
nfet_01v8
1 x 2 / 5
M14
nfet_01v8
1 x 2 / 5

The FVF



The CCM (without FVF)

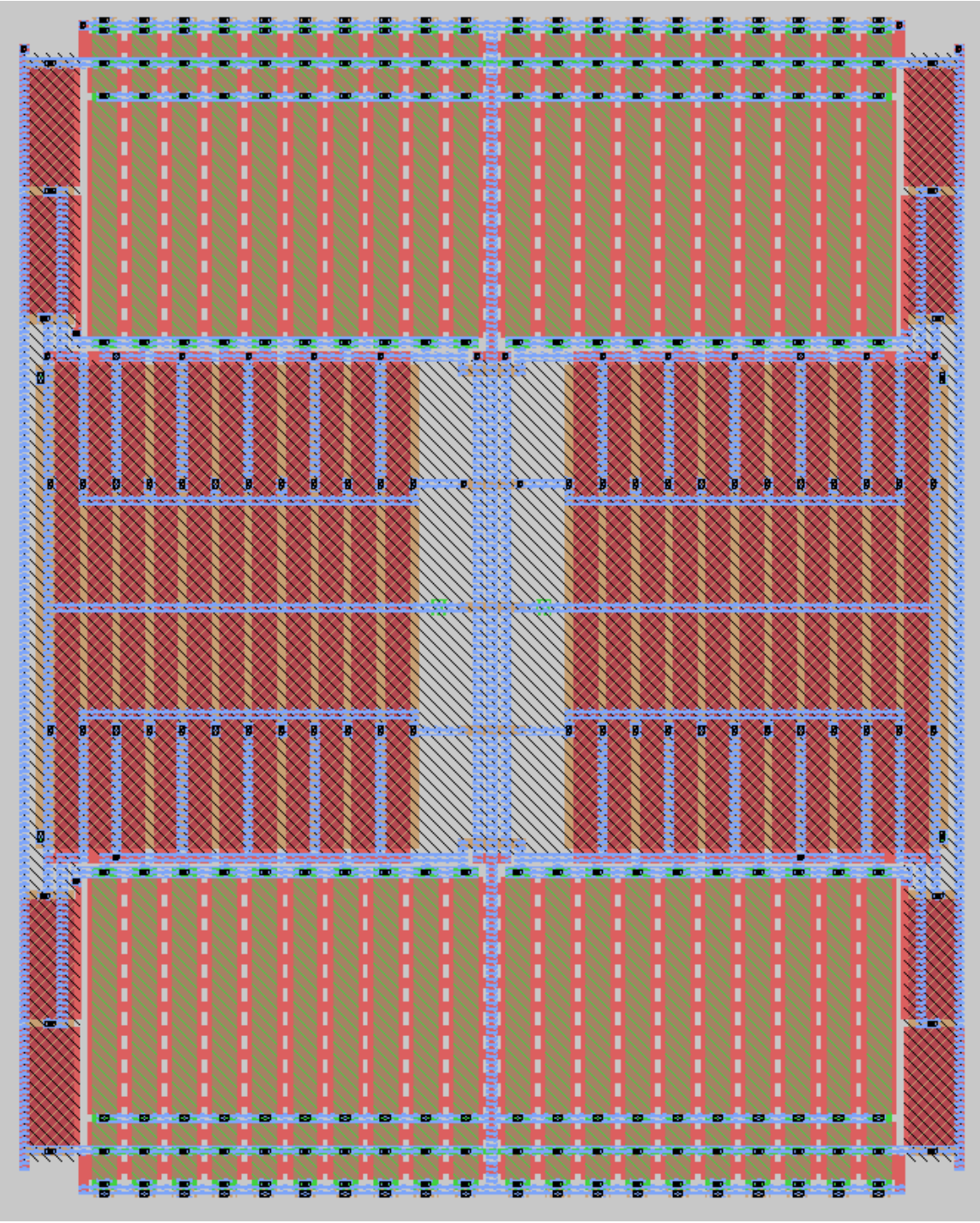
III. Layout Design

Here's a running list of commands I'm saving for future reference.

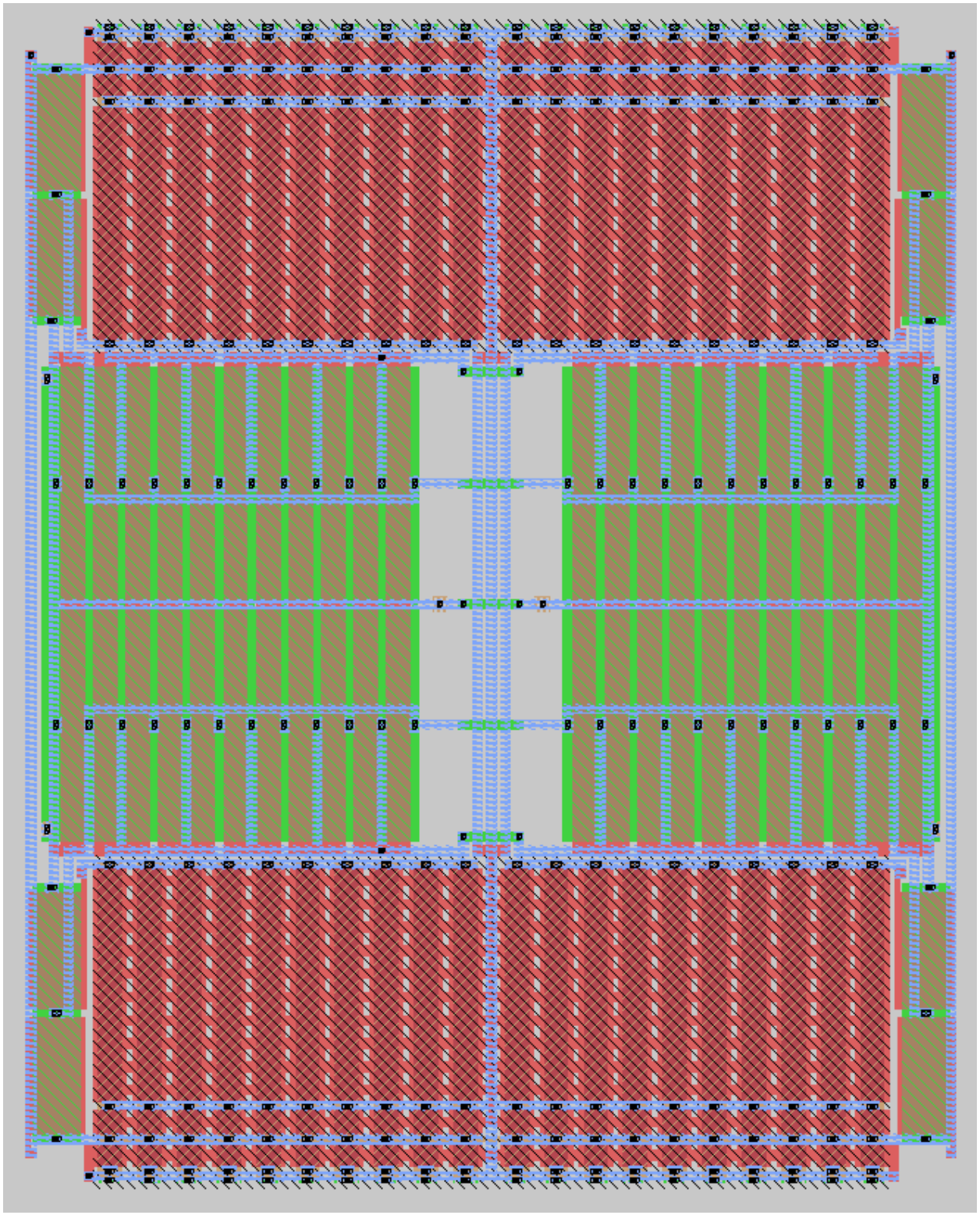
- On startup:
  - grid 0.05um 0.05um
  - snap user
  - drc style drc(full)
- Important commands:
  - save {cell_name} - must be used otherwise Segmentation fault (.mag is optional)
  - load {cell_name}
- On export:
  - extract all
  - ext2spice hierarchy on
  - ext2spice scale off
  - ext2spice cthresh infinite
  - ext2spice subcircuit top off
  - ext2spice
- General commands:
  - b - view box size
  - p - paint
  - s - highlight what is being hovered or extend highlight to electrically connected
  - : - start typing in command line
  - u - undo/U - redo
  - e - edit selected cell
  - tool - switch tool
  - select area - select all items intersecting selection area
  - move [dir] [dist] - move selection by dir/dist or to cursor (bottom left alignment)
  - copy [dir] [dist]
  - erase {item} - remove things of that type in the selection
  - label {name} [dir] [layer] - make a label
  - port make {num} - assign port to label
  - getcell {cell_name} - import a cell into design
  - drc why - check DRC errors
- Types of elements:
  - li - local interconnect
  - poly - polysilicon
    - pc - polysilicon contact (connects poly-li)
  - ndiff - n diffusion (for NMOS)
    - ndc - n diffusion contact (connects ndiff-li)
  - nwell - well for PMOS

- pdiff - p diffusion (for PMOS)
  - pdc - p diffusion contact (connects pdiff-li)
- psd - p+ substrate diffusion (for NMOS well contacts)
  - psc - p+ substrate contact (connects psd-li)
- nsd - n+ substrate diffusion (for PMOS well contacts)
  - nsc - n+ substrate contact (connects nsd-li)
- m1 - metal layer 1
  - mcon - metal contact (connects li-m1)

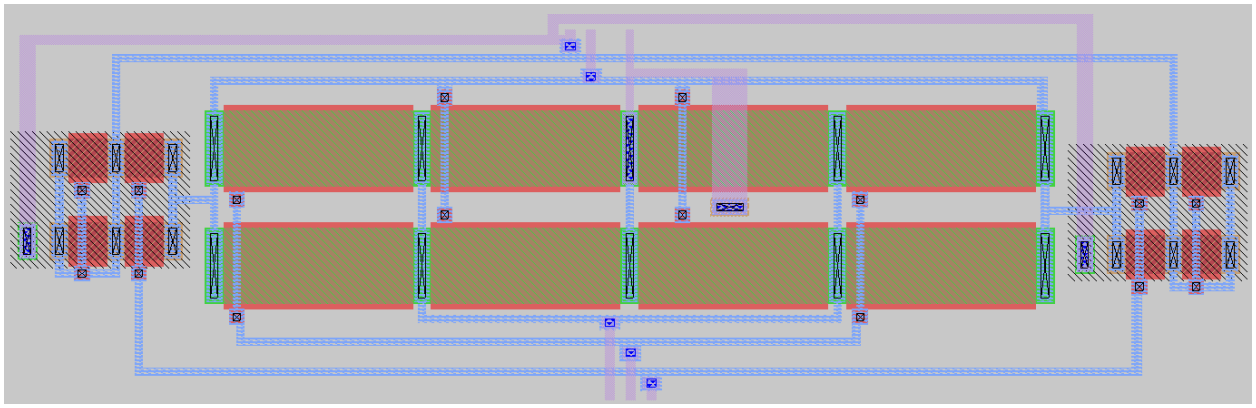
The layout was done based on the LDS with the same component structure as presented before. Here are the highlights.



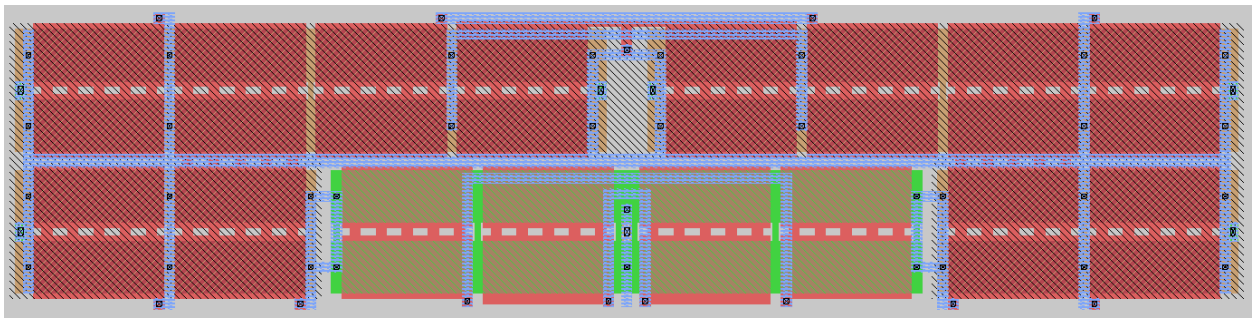
The PCBC



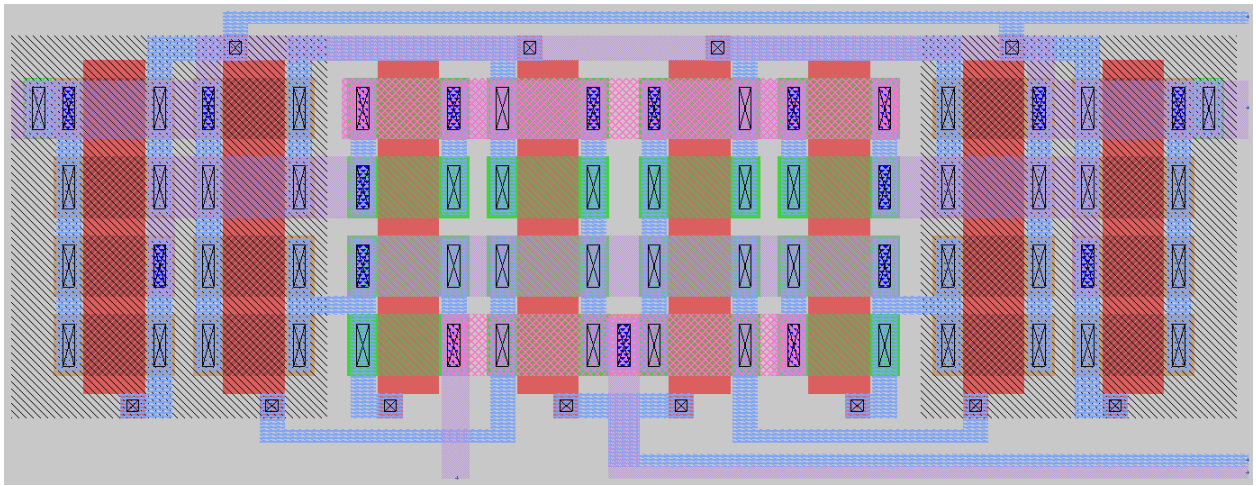
The NCBC



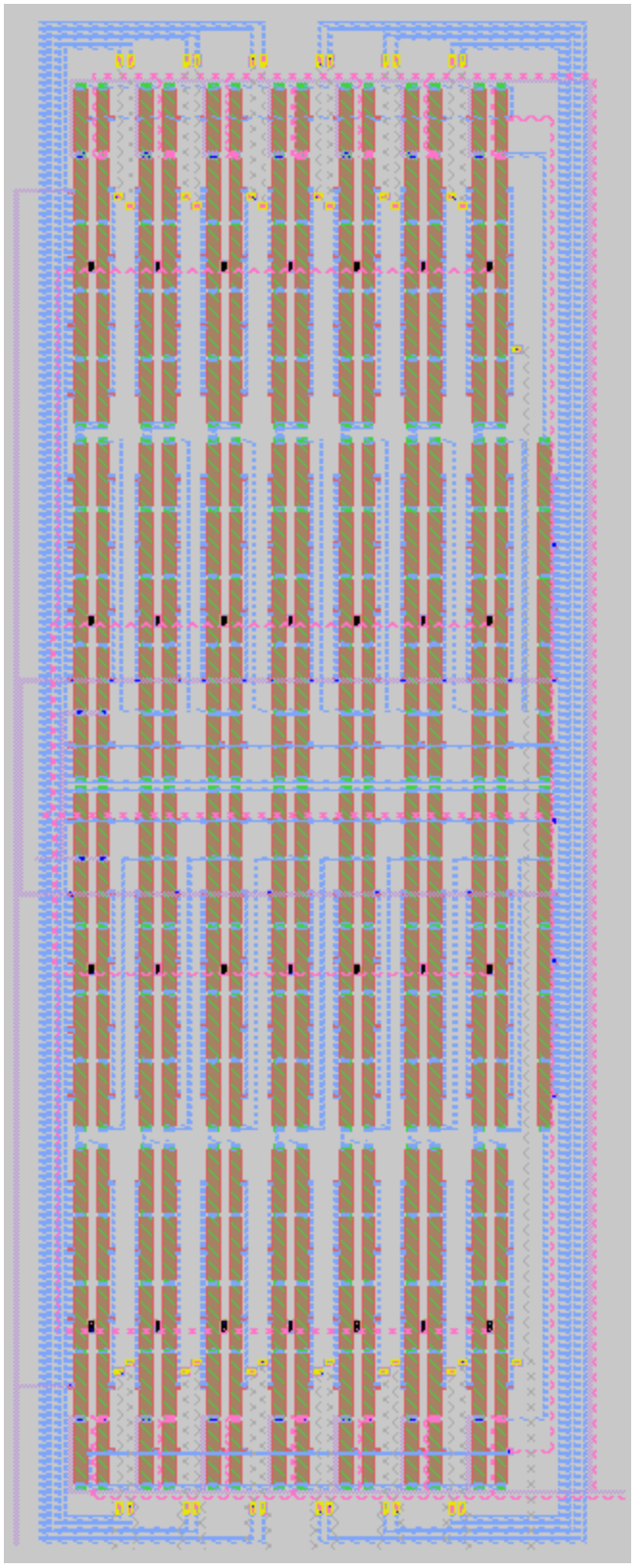
The FVF



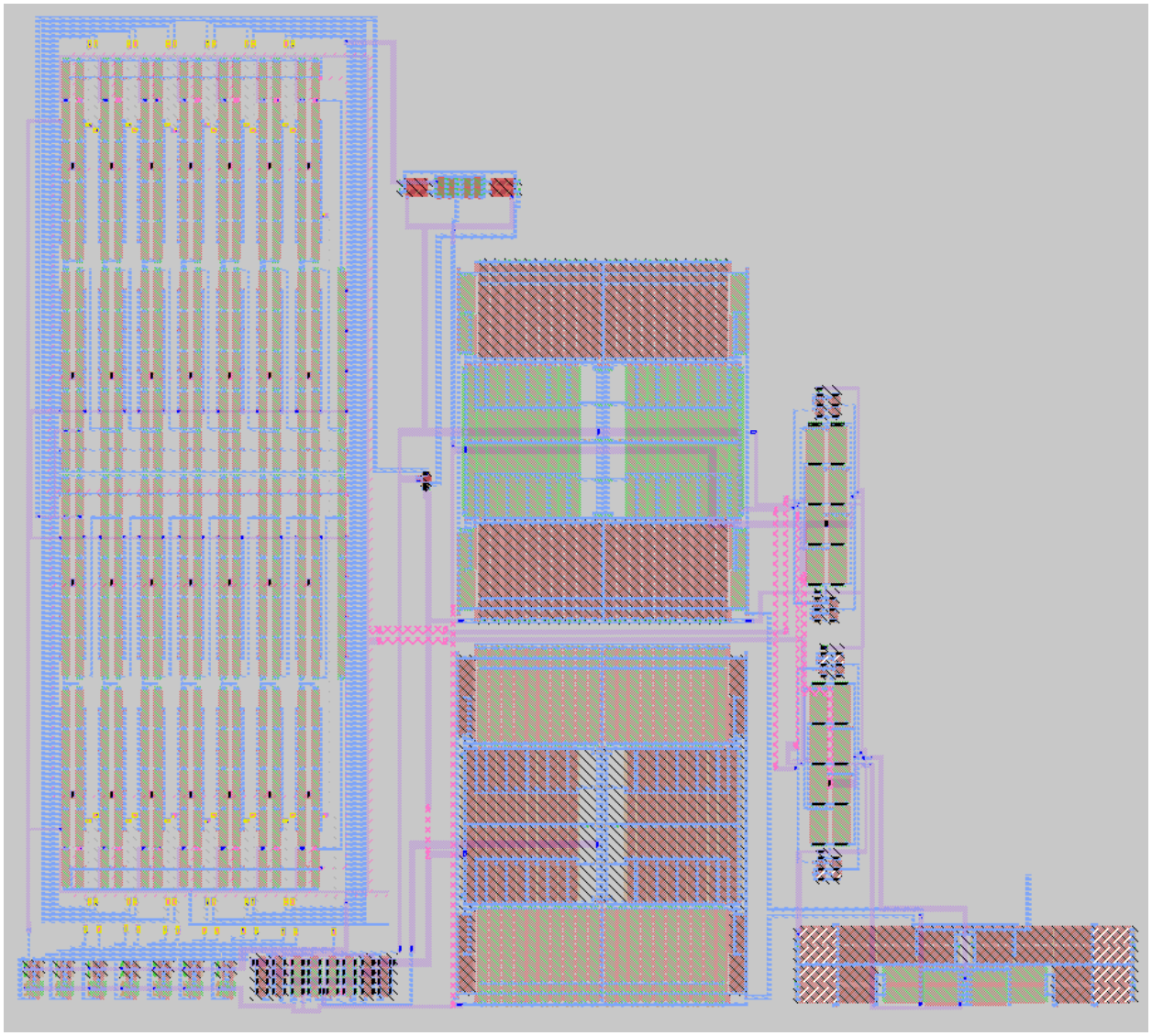
The CCM



The BBG



The DAC



Full Layout

IV. Layout vs Schematic

Throughout the process for each subcircuit and the final, the two netlists from the tools were run through Netgen in order to compare their nodes and edges. They matched uniquely, indicating that these two implementations are in fact identical.

To get rid of the error about the setup.tcl file being missing I had to copy it into my folder using the following command:

- cp /usr/local/share/pdk/sky130A/libs.tech/netgen/sky130A_setup.tcl ./setup.tcl

The output is below:

```
loading history file ... 48 events added
Running NetGen Console Functions
Netgen 1.5.299 compiled on Sat Sep  6 02:50:48 PM EDT 2025
Warning: netgen command 'format' use fully-qualified name
'::netgen::format'
Warning: netgen command 'global' use fully-qualified name
'::netgen::global'
Main console display active (Tcl8.6.14 / Tk8.6.14)
(layout) 49 % lvs LDS_mosdac.spice full.spice
Reading netlist file LDS_mosdac.spice
Call to undefined subcircuit LDS_bbg
Creating placeholder cell definition.
Call to undefined subcircuit LDS_ccm
Creating placeholder cell definition.
Call to undefined subcircuit LDS_ncbc
Creating placeholder cell definition.
Call to undefined subcircuit LDS_pcbc
Creating placeholder cell definition.
Call to undefined subcircuit LDS_fvf
Creating placeholder cell definition.
Call to undefined subcircuit LDS_dac
Creating placeholder cell definition.
Call to undefined subcircuit sky130_fd_pr__pfet_01v8
Creating placeholder cell definition.
Call to undefined subcircuit sky130_fd_pr__nfet_01v8
Creating placeholder cell definition.
Call to undefined subcircuit LDS_inverter
Creating placeholder cell definition.
Call to undefined subcircuit LDS_ccm_nofvf
Creating placeholder cell definition.
Reading netlist file full.spice
Call to undefined subcircuit sky130_fd_pr__nfet_01v8
Creating placeholder cell definition.
Call to undefined subcircuit sky130_fd_pr__pfet_01v8
Creating placeholder cell definition.

Reading setup file setup.tcl

Model sky130_fd_pr__nfet_01v8 pin 1 == 3
No property area found for device sky130_fd_pr__nfet_01v8
```

```
No property perim found for device sky130_fd_pr__nfet_01v8
No property topography found for device sky130_fd_pr__nfet_01v8
Model sky130_fd_pr__nfet_01v8 pin 1 == 3
No property mult found for device sky130_fd_pr__nfet_01v8
No property sa found for device sky130_fd_pr__nfet_01v8
No property sb found for device sky130_fd_pr__nfet_01v8
No property sd found for device sky130_fd_pr__nfet_01v8
No property nf found for device sky130_fd_pr__nfet_01v8
No property nrd found for device sky130_fd_pr__nfet_01v8
No property nrs found for device sky130_fd_pr__nfet_01v8
No property area found for device sky130_fd_pr__nfet_01v8
No property perim found for device sky130_fd_pr__nfet_01v8
No property topography found for device sky130_fd_pr__nfet_01v8
Model sky130_fd_pr__pfet_01v8 pin 1 == 3
No property area found for device sky130_fd_pr__pfet_01v8
No property perim found for device sky130_fd_pr__pfet_01v8
No property topography found for device sky130_fd_pr__pfet_01v8
Model sky130_fd_pr__pfet_01v8 pin 1 == 3
No property mult found for device sky130_fd_pr__pfet_01v8
No property sa found for device sky130_fd_pr__pfet_01v8
No property sb found for device sky130_fd_pr__pfet_01v8
No property sd found for device sky130_fd_pr__pfet_01v8
No property nf found for device sky130_fd_pr__pfet_01v8
No property nrd found for device sky130_fd_pr__pfet_01v8
No property nrs found for device sky130_fd_pr__pfet_01v8
No property area found for device sky130_fd_pr__pfet_01v8
No property perim found for device sky130_fd_pr__pfet_01v8
No property topography found for device sky130_fd_pr__pfet_01v8
Comparison output logged to file comp.out
Logging to file "comp.out" enabled
Circuit sky130_fd_pr__pfet_01v8 contains no devices.
Circuit sky130_fd_pr__nfet_01v8 contains no devices.

Contents of circuit 1:  Circuit: 'LDS_mosdac.spice'
Circuit LDS_mosdac.spice contains 771 device instances.
  Class: sky130_fd_pr__nfet_01v8 instances: 524
  Class: sky130_fd_pr__pfet_01v8 instances: 247
Circuit contains 243 nets.
Contents of circuit 2:  Circuit: 'full.spice'
Circuit full.spice contains 771 device instances.
  Class: sky130_fd_pr__nfet_01v8 instances: 524
  Class: sky130_fd_pr__pfet_01v8 instances: 247
Circuit contains 243 nets.
```

```
Circuit was modified by parallel/series device merging.
New circuit summary:

Contents of circuit 1:  Circuit: 'LDS_mosdac.spice'
Circuit LDS_mosdac.spice contains 307 device instances.
  Class: sky130_fd_pr__nfet_01v8 instances: 277
  Class: sky130_fd_pr__pfet_01v8 instances:  30
Circuit contains 243 nets.
Contents of circuit 2:  Circuit: 'full.spice'
Circuit full.spice contains 307 device instances.
  Class: sky130_fd_pr__nfet_01v8 instances: 277
  Class: sky130_fd_pr__pfet_01v8 instances:  30
Circuit contains 243 nets.

Circuit 1 contains 307 devices, Circuit 2 contains 307 devices.
Circuit 1 contains 243 nets,    Circuit 2 contains 243 nets.

No more changes can be made to series/parallel networks.

Final result:
Circuits match uniquely.
Property errors were found.

The following cells had property errors:
 LDS_mosdac.spice

Logging to file "comp.out" disabled
LVS Done.
```

This can also be confirmed in the comp.out file (contents below for reference):

```
Circuit 1 cell sky130_fd_pr__pfet_01v8 and Circuit 2 cell
sky130_fd_pr__pfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__pfet_01v8 and sky130_fd_pr__pfet_01v8 are
equivalent.

Circuit 1 cell sky130_fd_pr__nfet_01v8 and Circuit 2 cell
sky130_fd_pr__nfet_01v8 are black boxes.
Equate elements:  no current cell.
Device classes sky130_fd_pr__nfet_01v8 and sky130_fd_pr__nfet_01v8 are
equivalent.
Flattening unmatched subcell LDS_bbg in circuit LDS_mosdac.spice (0)(1
instance)
Flattening unmatched subcell LDS_ccm in circuit LDS_mosdac.spice (0)(1
instance)
Flattening unmatched subcell LDS_ccm_nofvf in circuit LDS_mosdac.spice
(0)(1 instance)
Flattening unmatched subcell LDS_fvf in circuit LDS_mosdac.spice (0)(2
instances)
Flattening unmatched subcell LDS_ncbc in circuit LDS_mosdac.spice (0)(1
instance)
Flattening unmatched subcell LDS_pcbc in circuit LDS_mosdac.spice (0)(1
instance)
Flattening unmatched subcell LDS_dac in circuit LDS_mosdac.spice (0)(1
instance)
Flattening unmatched subcell LDS_inverter in circuit LDS_mosdac.spice (0)(7
instances)

Class LDS_mosdac.spice (0):  Merged 464 parallel devices.
Class full.spice (1):  Merged 464 parallel devices.
Subcircuit summary:
Circuit 1: LDS_mosdac.spice                |Circuit 2: full.spice
-------------------------------------------|------------------------------
------------
sky130_fd_pr__pfet_01v8 (247->30)          |sky130_fd_pr__pfet_01v8
(247->30)
sky130_fd_pr__nfet_01v8 (524->277)         |sky130_fd_pr__nfet_01v8
(524->277)
Number of devices: 307                     |Number of devices: 307
Number of nets: 243                        |Number of nets: 243
```

```
--------------------------------------------------------------------------------
------------
Resolving symmetries by property value.
Resolving symmetries by pin name.
Resolving symmetries by net name.
Netlists match with 232 symmetries with property errors.
sky130_fd_pr__pfet_01v8:M3 vs. sky130_fd_pr__pfet_01v8:760:
 W circuit1: 0.45   circuit2: 0.55   (delta=20%, cutoff=1%)
Cells have no pins;  pin matching not needed.
Device classes LDS_mosdac.spice and full.spice are equivalent.

Final result: Circuits match uniquely.
Property errors were found.

The following cells had property errors:
 LDS_mosdac.spice
```