SSTL Circuit Design

(Derek Hines-Mohrman, Bespoke Silicon Group)

Github: https://github.com/bespoke-silicon-group/bsg_ddr3_io

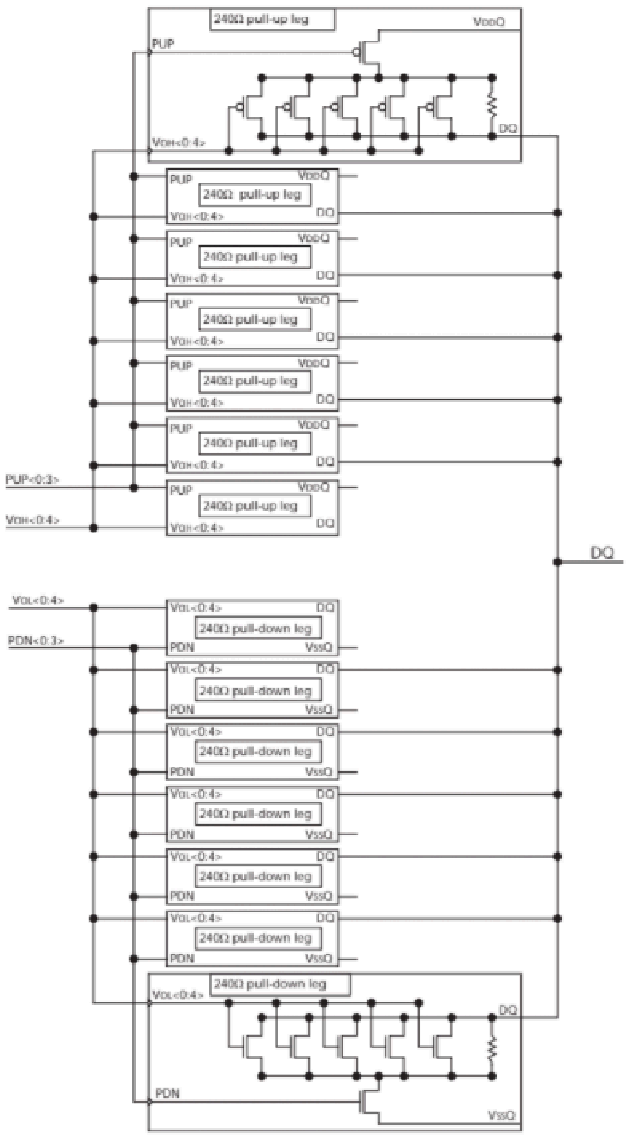
Overview

This guide attempts to walk the reader through the process of designing a bidirectional SSTL I/O suitable for DDR3 DRAM, and assumes the reader has taken a basic VLSI course. The simulation infrastructure assumes that spice models that are available that are compatible with NGSPICE, such as in the open source SKY130 process. This document is a work in progress, is intended purely as a learning tool, and we make no warranties about the suitability of this information.

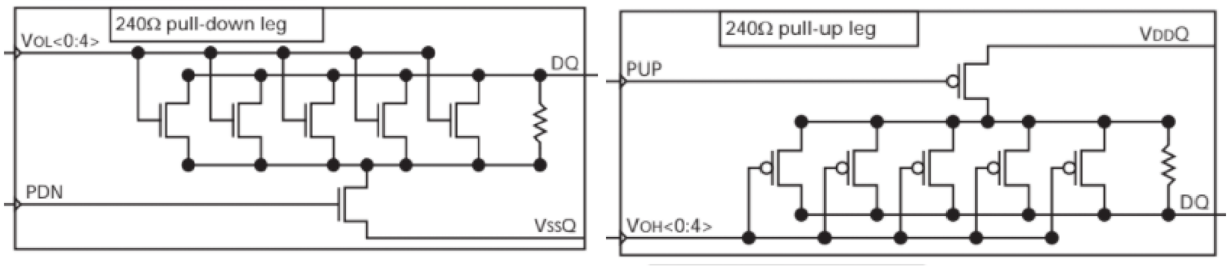
The implemented DDR3 SSTL circuit serves as both the output driver, and on die termination (ODT) for receiving data on the data line “DQ”. This design does not implement the differential comparator or level shifter that would ordinarily be used to resolve incoming signals to a 0 or a 1. (Our test chip uses a standard inverter as the receiver, but this is not standard practice.) SKY130 uses a nominal voltage of 1.8V, which is slightly above the target of 1.5V in SSTL, so we use the standard CMOS transistors in the process at slightly below the nominal voltage. We assume that an analog I/O (with clamps) is available for the process. The SSTL is a necessary part of a DDR3 memory controller. It is the driver circuit for the data lines, and also serves as the termination when the controller is receiving data. Ideally, this cell would go under the IO pads for the data (DQ) pins for the memory interface.

Each SSTL driver contains 7 pull-up and 7 pull-down "legs". All 14 legs are in parallel. Each leg can be considered to be a controllable resistor. When enabled, a pullup leg will connect the DQ pin through a 240 Ω resistor to VDD. So the legs pulls up the DQ pin with 240 Ω impedance. When disabled, a pullup leg disconnects from the DQ pin (the leg makes a high impedance connection). Similarly, when enabled, a pulldown leg will connect the DQ pin through a 240 Ω resistor to VSS. Disabling a pulldown leg disconnects it from DQ.

This design process was used with the skywater 130 nm open source PDK.



SSTL block diagram



Detail diagrams of pull-down and pull-up legs

Enabling and disabling legs is how the driver is switched from transmitting to receiving mode.

When driving DQ, the SSTL has some number of legs enabled on one side (up or down) and none of the legs enabled on the other side. So when driving DQ high, some number of pullup legs are enabled, and none of the pulldown legs are enabled. The specific number of pullup legs enabled controls the output impedance of the signal. When driving DQ low, the only pulldown legs are enabled.

The DDR3 specification requires driving impedances to be configurable to 40 Ω and 34.29 Ω. The is achieved by the SSTL by enabling either 6 or 7 legs respectively.

When receiving, some number of legs are enabled on both the pullup and pulldown sides. This means the SSTL is actually driving DQ to the voltage VDD/2. It is expected the driving circuit on the other end has a much lower impedance so it can effectively control the voltage of DQ.

The DDR3 specification requires termination impedances to be configurable to 120 Ω, 60 Ω, and 40 Ω. This achieved by SSTL by enabling (1 pullup and 1 pulldown), (2 pullup and 2 pulldown), and (3 pullup and 3 pulldown) legs respectively.

The additional feature an SSTL must have is the ability to fine tune the resistances of every leg to keep the resistance close to the required 240 Ω. The DDR3 spec includes a calibration procedure where the true SSTL drive strength is measured periodically. If it falls out of range, the leg resistances are adjusted. In this particular SSTL implementation, there are several calibration FETs in parallel with the main (polysilicon) resistor in each leg. In the low temperature or high voltage cases (where the leg has reduced resistance) some calibration FETs are turned off to increase the resistance back in spec. In the high temperature or low voltage cases, more calibration FETs are turned on.

Requirements

Requirements are derived from the DDR3 specification: JESD79-3E

Voltages

V_{SSQ} = 0V, V_{DDQ} = 1.5 V

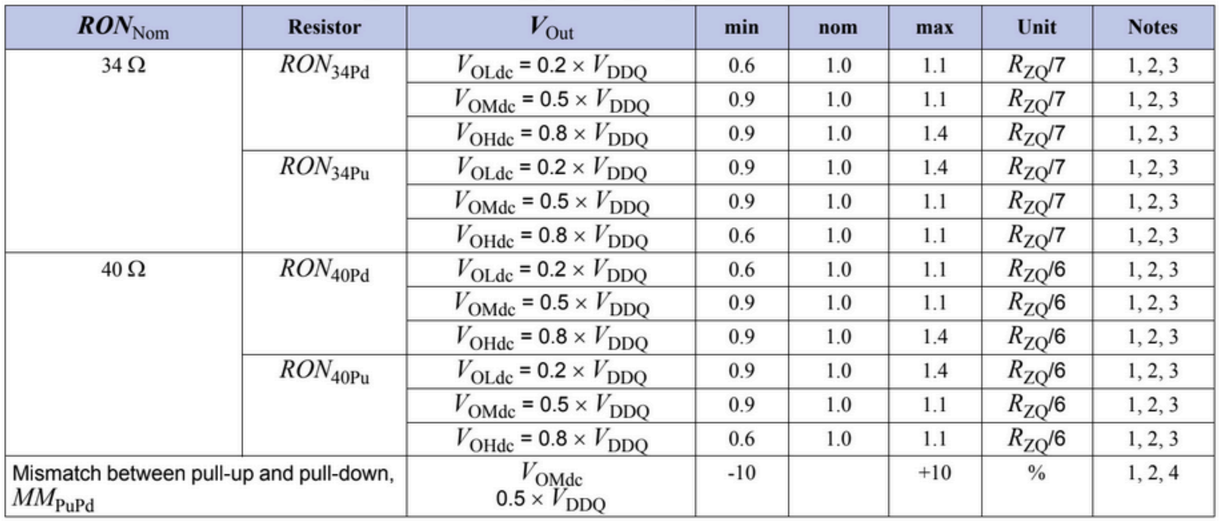
To operate properly, supply voltage V_{DD} must be between 1.475V and 1.575V.

The absolute maximum values of V_{DD} must be between -0.4V and 1.975V (relative to V_{SS}) (JESD79-3E, pg 109)

Resistance

After calibration, each leg resistance must be with a certain range of the target resistance of 240Ω. The resistance is measured at 3 levels of V_{DQ}: 0.2, 0.5, and 0.8 x V_{DD} (typically 1.5V). The resistance must fall in range as per the table from the spec:

Table 38 — Output Driver DC Electrical Characteristics, assuming R_{ZQ} = 240 Ω ; entire operating temperature range; after proper ZQ calibration



RON_{Nom}	Resistor	V_{Out}	min	nom	max	Unit	Notes
34 Ω	RON_{34Pd}	$V_{OLdc} = 0.2 \times V_{DDQ}$	0.6	1.0	1.1	$R_{ZQ}/7$	1, 2, 3
		$V_{OMdc} = 0.5 \times V_{DDQ}$	0.9	1.0	1.1	$R_{ZQ}/7$	1, 2, 3
		$V_{OHdc} = 0.8 \times V_{DDQ}$	0.9	1.0	1.4	$R_{ZQ}/7$	1, 2, 3
	RON_{34Pu}	$V_{OLdc} = 0.2 \times V_{DDQ}$	0.9	1.0	1.4	$R_{ZQ}/7$	1, 2, 3
		$V_{OMdc} = 0.5 \times V_{DDQ}$	0.9	1.0	1.1	$R_{ZQ}/7$	1, 2, 3
		$V_{OHdc} = 0.8 \times V_{DDQ}$	0.6	1.0	1.1	$R_{ZQ}/7$	1, 2, 3
40 Ω	RON_{40Pd}	$V_{OLdc} = 0.2 \times V_{DDQ}$	0.6	1.0	1.1	$R_{ZQ}/6$	1, 2, 3
		$V_{OMdc} = 0.5 \times V_{DDQ}$	0.9	1.0	1.1	$R_{ZQ}/6$	1, 2, 3
		$V_{OHdc} = 0.8 \times V_{DDQ}$	0.9	1.0	1.4	$R_{ZQ}/6$	1, 2, 3
	RON_{40Pu}	$V_{OLdc} = 0.2 \times V_{DDQ}$	0.9	1.0	1.4	$R_{ZQ}/6$	1, 2, 3
		$V_{OMdc} = 0.5 \times V_{DDQ}$	0.9	1.0	1.1	$R_{ZQ}/6$	1, 2, 3
		$V_{OHdc} = 0.8 \times V_{DDQ}$	0.6	1.0	1.1	$R_{ZQ}/6$	1, 2, 3
Mismatch between pull-up and pull-down, MM_{PuPd}		V_{OMdc} $0.5 \times V_{DDQ}$	-10		+10	%	1, 2, 4

Note the symmetry between the pull-up and pull-down legs. It can be seen the resistance is allowed to grow or reduce along with the voltage across the leg. Also note the requirements are identical for the 34Ω and 40Ω case. This makes sense if we have 7 pull-up and 7 pull-down legs. These requirements directly translate to each leg with resistance R_{ZQ} as labeled in the table. (JESD79-3E, pg 129, 132-133)

These are the derived resistance requirements for each leg:

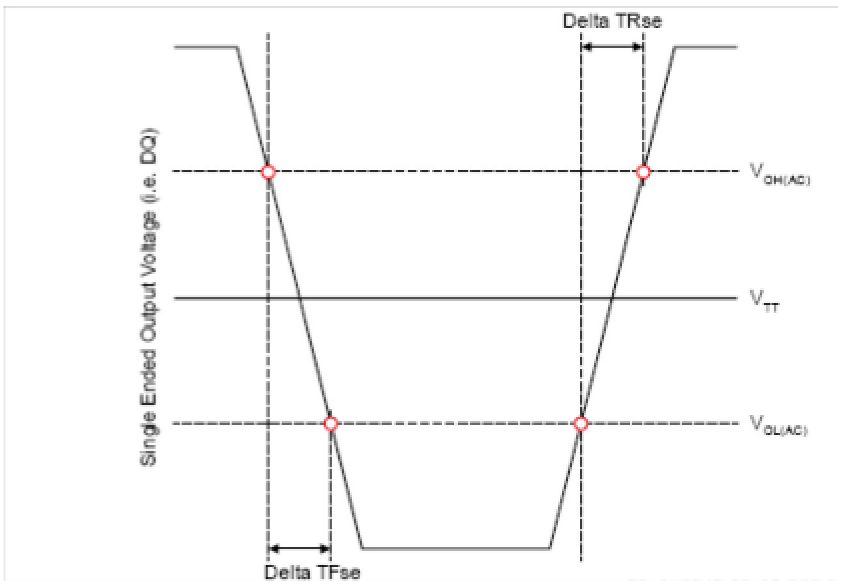
Voltage across leg (V)	**Minimum allowed Resistance (Ω)**	**Maximum allowed Resistance (Ω)**
0.3	144	264
0.75	216	264
1.2	216	336

This requirement should be met AFTER calibration.

Slew Rate

The required slew rate for V_{DQ} is between 2.5 and 5 V/ns for the single ended case. For the differential case, the required slew rate is between 5 and 10 V/ns. (JESD79-3E, pg 123-125) (Which makes sense because V_{DQ}# is defined differentially as well, so since both lines are changing voltage towards each other, the slew will be twice as fast.)

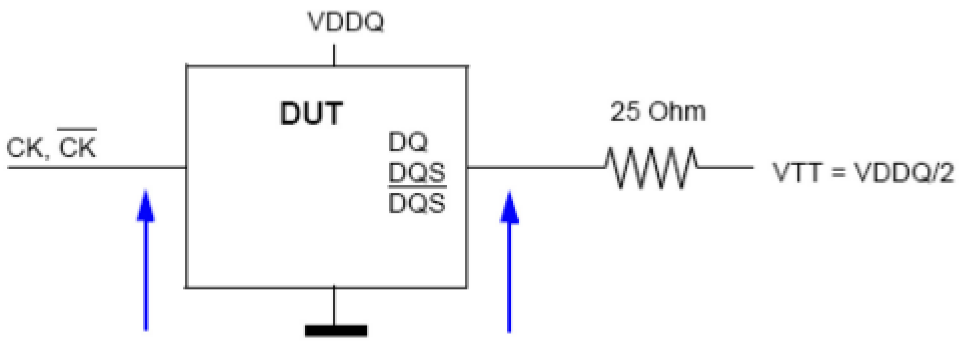
Slew rate is measured as the time it takes to pull up or down V_{DQ} between $0.4*V_{DDQ}$ and $0.6*V_{DDQ}$.



Single ended slew rate definition

This requirement is specifically for the “RZQ/7 setting” which means ALL of the pull up and pull down legs are being toggled to cause this voltage transition.

The slew rate should be measured with a 25Ω output resistor tied to $V_{DDQ}/2$.



Slew Measurement Simulation Setup

Capacitance

The input and output capacitance of the device should always be between 1.4pF and 2.1pF to meet the highest speed variant of DDR3 (DDR3-2133). The most lenient variant requires a capacitance range of 1.3pF to 3.0pF (DDR3-800) (JESD79-3E, pg 154-155)

Temperature

“Normal” temperature range is 0C to 85C. There is an extended temperature range as well up to 95C. (JESD79-3E, pg 109)

When designing for sky130, we chose to support temperatures up to 125C.

Calibration

Sources of error in resistance to be calibrated for:

- Temperature (-40C to 125C)
- Process (capacitance of FET gates, output resistance of FETs.)
- V_{DD} (1.5V I/O voltage)

Note that change in V_{DQ} cannot be calibrated for, (this is the voltage we are pulling up/down!)

Design flow

We developed an open-source flow found here to aid in the design and simulation of this circuit.

https://github.com/bespoke-silicon-group/bsg_ddr3_io

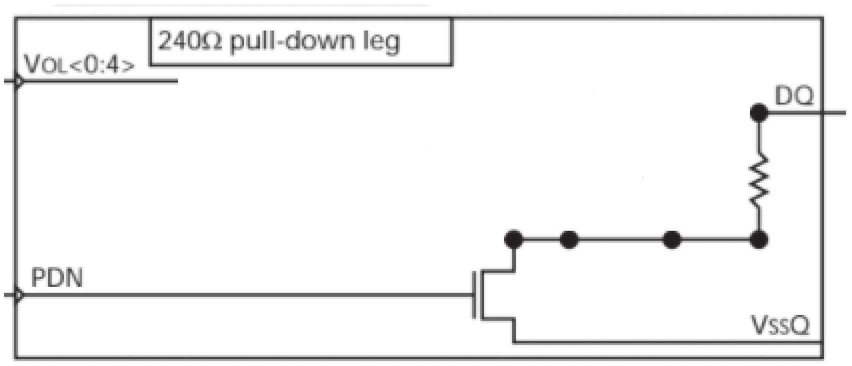
For initial design, we used the circuit schematic tool XSCHEM.

Simulations are performed with NGSPICE.

The layout of the circuit was done with MAGIC Layout. This tool was also used for the parasitic extraction needed for the post-layout spice simulations.

Step 1: Size resistor and main control FET

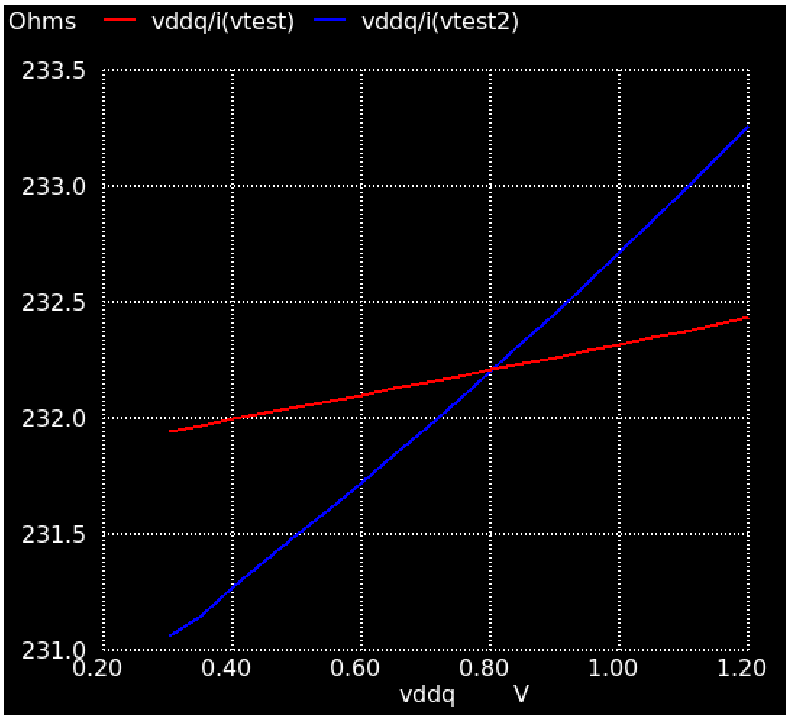
In Step 1, the goal is to characterize the approximate size of the resistor and main pull up/down FET needed.



Simplified Pull Down leg for Step 1

When designing for sky130, a polysilicon resistor for both the pull up and pull down legs.

One tradeoff we need to make is how much of the resistance of the leg should come from the poly resistor, and how much from the FET? The lower the resistance of the FET, the larger we need to make it. But the larger the resistance of the FET, the more the resistance will vary as V_{DQ} changes. This means we will have less margin for error in the resistance, and will need to calibrate more precisely.



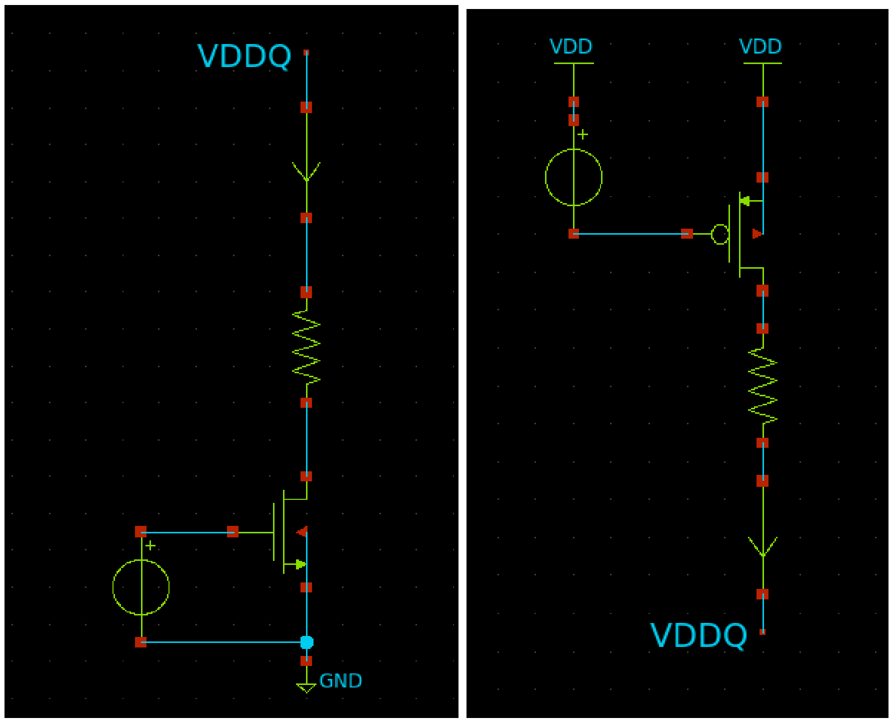
Example: Low res FET (red) vs high res FET (blue) resistance versus V_{DQ}

Step 2: Size resistor for min PVT

The calibration FETs (The ones in parallel with the poly resistor) can only take away from the resistance and never add to it. Therefore, we must find the worst case (minimum resistance) PVT corner for the legs, and then size the resistor so it is in valid range.

For the sky130 pdk, it was not obvious which process corner resulted in the lowest resistance. For completeness, all process corners were simulated.

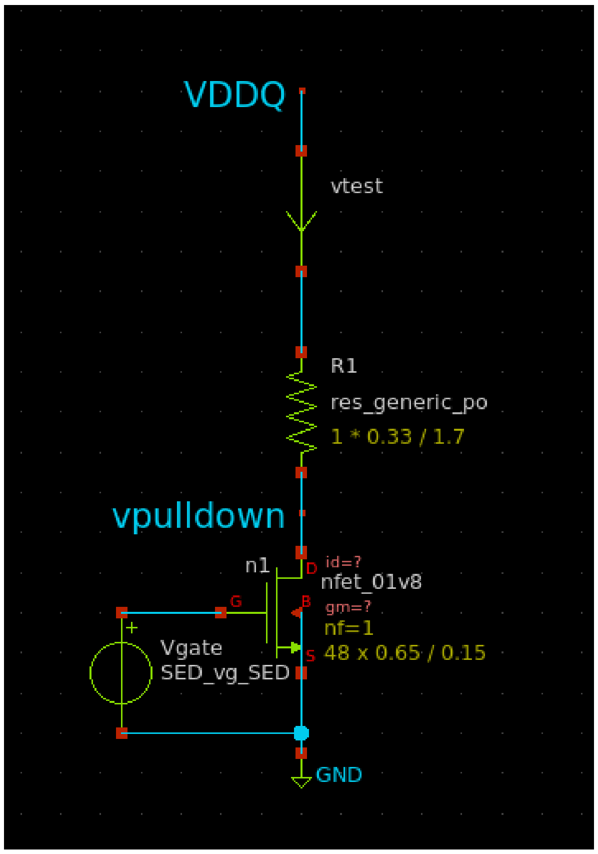
Below are the example circuits for testing the resistor and FET size:

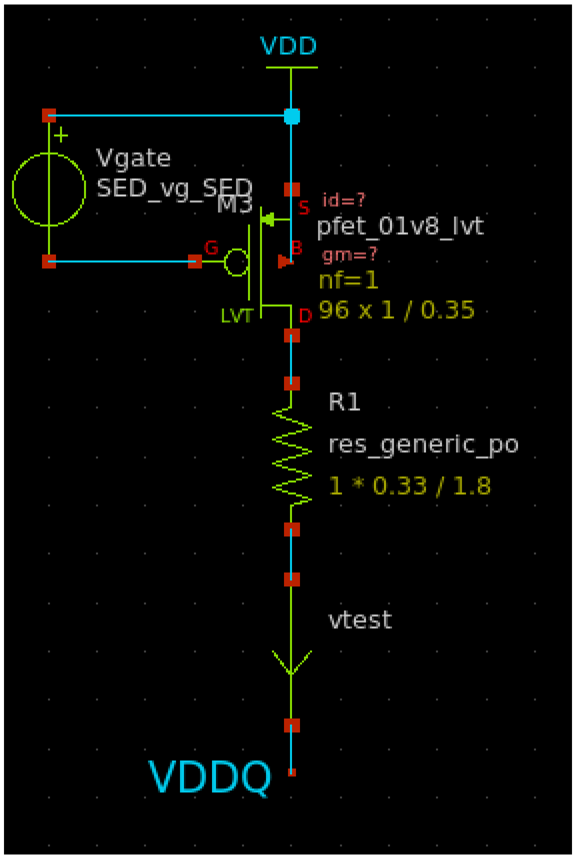


Resistor sizing circuit for pull-down leg (left) and pull-up leg(right)

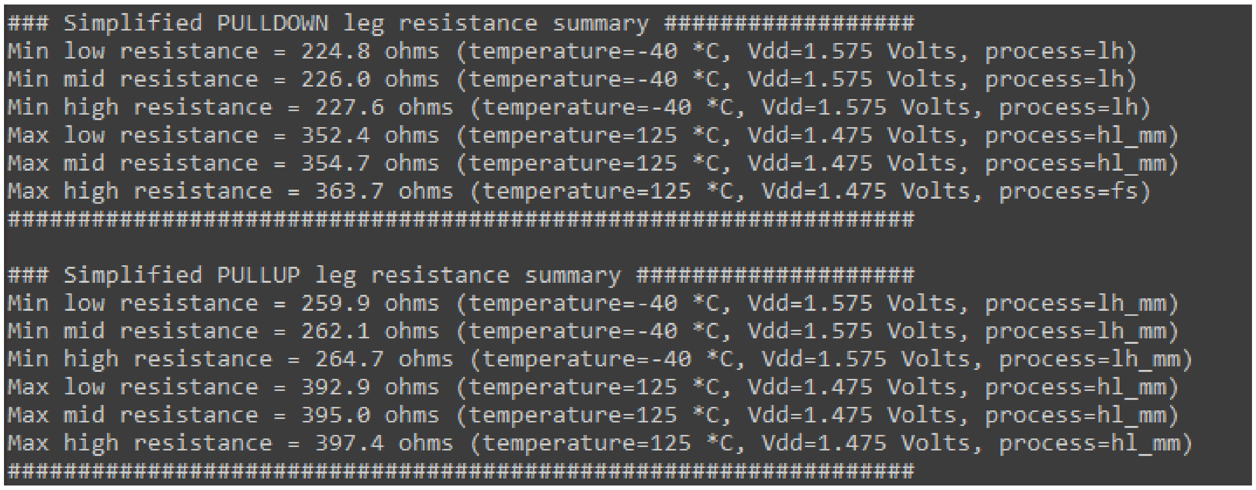
Results

After simulation, we choose the pull-down resistor to be **0.33um wide** (minimum width) by **1.7um long**. We choose the pull-up resistor to be **0.33um wide** by **1.8um long**. (In part, these values were chosen based on further simulations, but here we just present the final choice.) To set up the simulation for this, set the device properties for the resistors and FETs in the schematic file `schem/test_pd_res.sch` and `schem/test_pu_res.sch` .





To measure the max and min resistances at all corners, run the command `make simple-leg-sim`. The result of the simulation is shown below:



```
### Simplified PULLDOWN leg resistance summary ##################
Min low resistance = 224.8 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=lh)
Min mid resistance = 226.0 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=lh)
Min high resistance = 227.6 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=lh)
Max low resistance = 352.4 ohms (temperature=125 *C, Vdd=1.475 Volts, process=hl_mm)
Max mid resistance = 354.7 ohms (temperature=125 *C, Vdd=1.475 Volts, process=hl_mm)
Max high resistance = 363.7 ohms (temperature=125 *C, Vdd=1.475 Volts, process=fs)
#################################################################

### Simplified PULLUP leg resistance summary ####################
Min low resistance = 259.9 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=lh_mm)
Min mid resistance = 262.1 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=lh_mm)
Min high resistance = 264.7 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=lh_mm)
Max low resistance = 392.9 ohms (temperature=125 *C, Vdd=1.475 Volts, process=hl_mm)
Max mid resistance = 395.0 ohms (temperature=125 *C, Vdd=1.475 Volts, process=hl_mm)
Max high resistance = 397.4 ohms (temperature=125 *C, Vdd=1.475 Volts, process=hl_mm)
#################################################################
```

The ‘low’ resistances refer to the test corner where there is 0.3 V across the leg, the ‘mid’ resistances refer to the 0.75 V case, and the ‘high’ resistances refer to the 1.2 V case. The ‘min’ and ‘max’ for each test case refer to how the resistance varies *across PVT corners*.

For this first simulation, we care only about whether the ‘min’ resistance requirements are being met. And looking at the chart, we can see this is the case. The closest it comes to failing is the pullup ‘Min mid’ case where we only meet the 264 Ω requirement by 2 Ω.

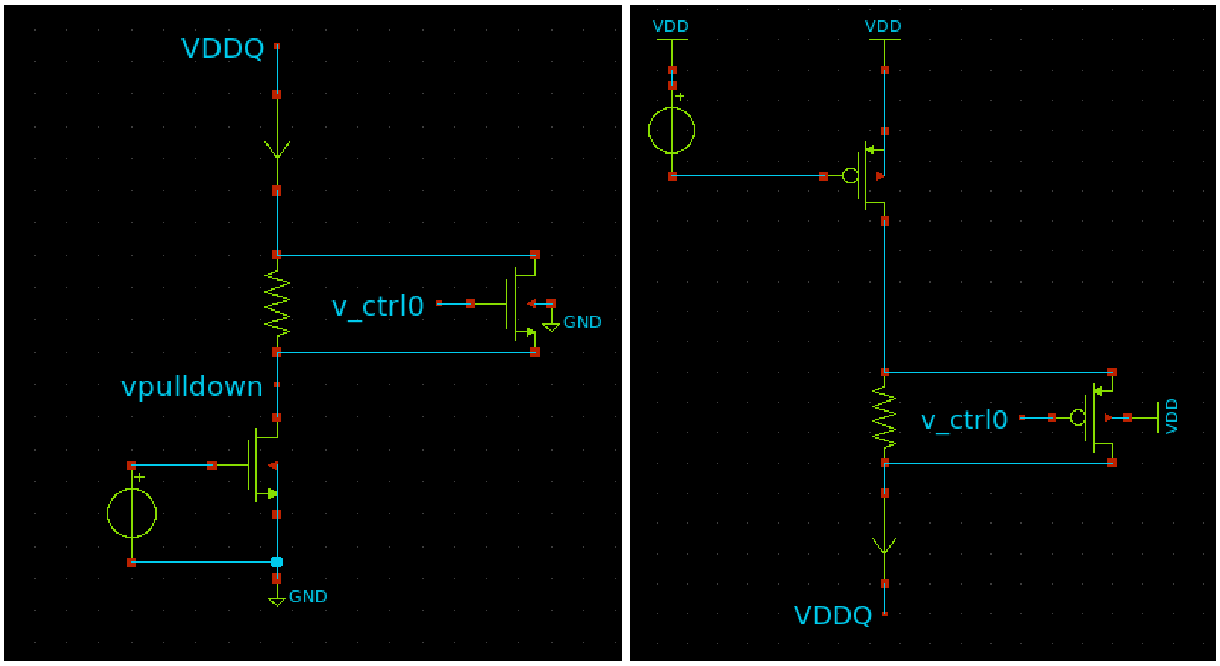
Step 3: Add single calibration FET

Now add one FET in parallel with the poly resistor.

The goal is so when the calibration FET is turned on (same gate voltage as the gate voltage of the main FET), the leg should satisfy the requirement in the maximum resistance PVT corner.

For this simulation, we need to find the maximum resistance corner case. Again, all process corners were tested to find the one resulting in the highest resistance.

In the following step, this fet will be split into several smaller FETs for the final leg design.



Single calibration FET sizing circuit for pull-down and pull-up legs

We did not set up a dedicated simulation for this step, instead we used the same simulation to test this intermediate design, and the final leg design from step 4.

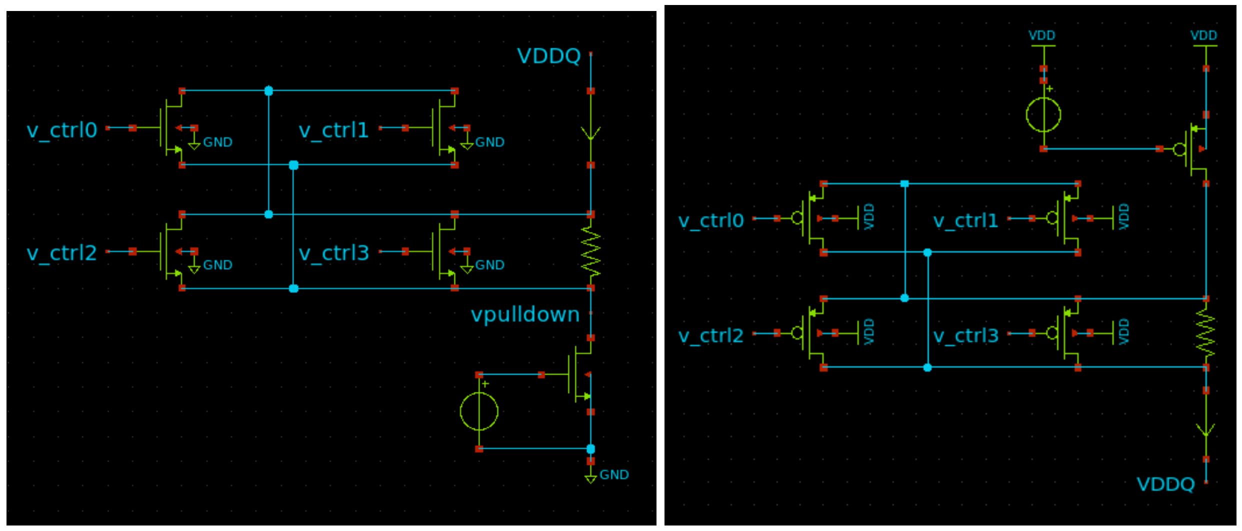
Step 4: Split calibration FET

Split the calibration FET into a number of smaller parallel FETs.

Start with 4 FETs. When they are all on, the leg should satisfy the requirement in the maximum resistance PVT corner just like in step 3.

The FETs should be sized approximately exponentially (powers of 2 for example) so we have the most calibration control with the fewest devices. (For example, each calibration FET is half the size of the next largest one). More than 4 calibration FETs may be needed, or possibly fewer. We would like to get away with as few as possible. While designing for sky130, I found that the calibration FETs worked best when each FET is about a third to a quarter the size of the previous FET.

The largest calibration FET (which should be about half the size of the calibration FET from step 3) is labeled with index 0. Larger index will mean a smaller calibration FET.



Complete pull-down and pull-up designs (4 calibration FETs)

Results

At this point we ran a new simulation testing the resistances of each leg type at all corners, when *all* calibration FETs are turned. This is to make sure the resistance will be low enough at the highest resistance PVT corner.

Also need to test that the 4 calibration FETs provide enough control to meet the resistance ranges at all PVT corners. This in particular was not tested until the SSTL calibration simulation was complete. But the final results are presented here:

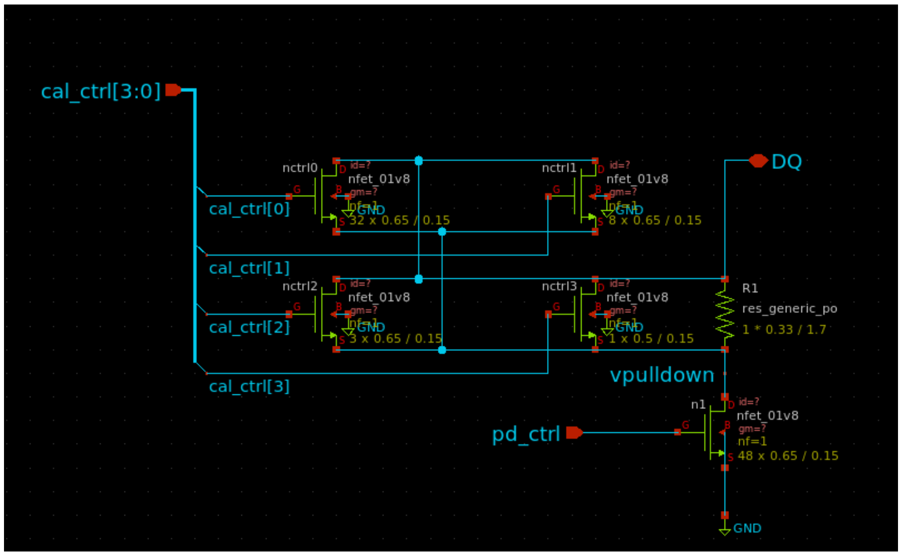
After simulating we choose these calibration FET sizes:

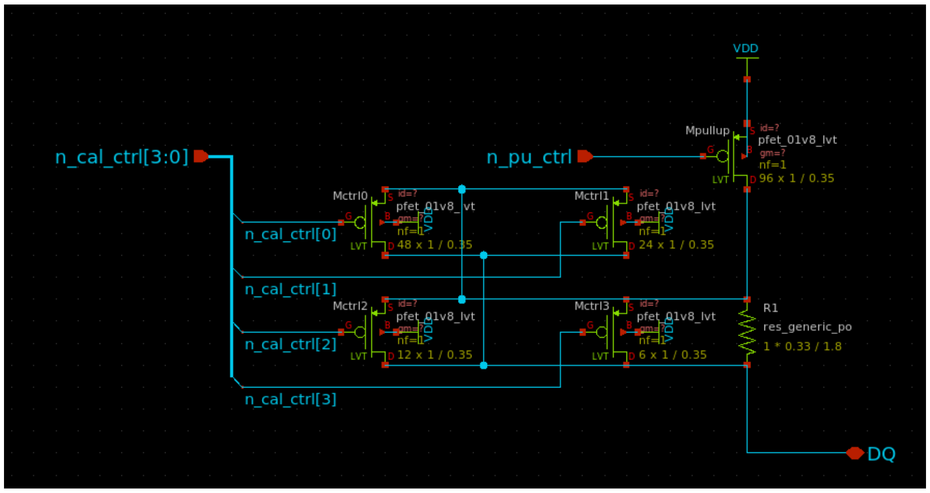
Device	multiple	Width (nm)	Length (nm)
Pull-down cal 0	32	650	150 (minimum)

Pull-down cal 1	8	650	150
Pull-down cal 2	3	650	150
Pull-down cal 3	1	500	150
Pull-up cal 0	48	1000	350 (minimum)
Pull-up cal 1	24	1000	350
Pull-up cal 2	12	1000	350
Pull-up cal 3	6	1000	350

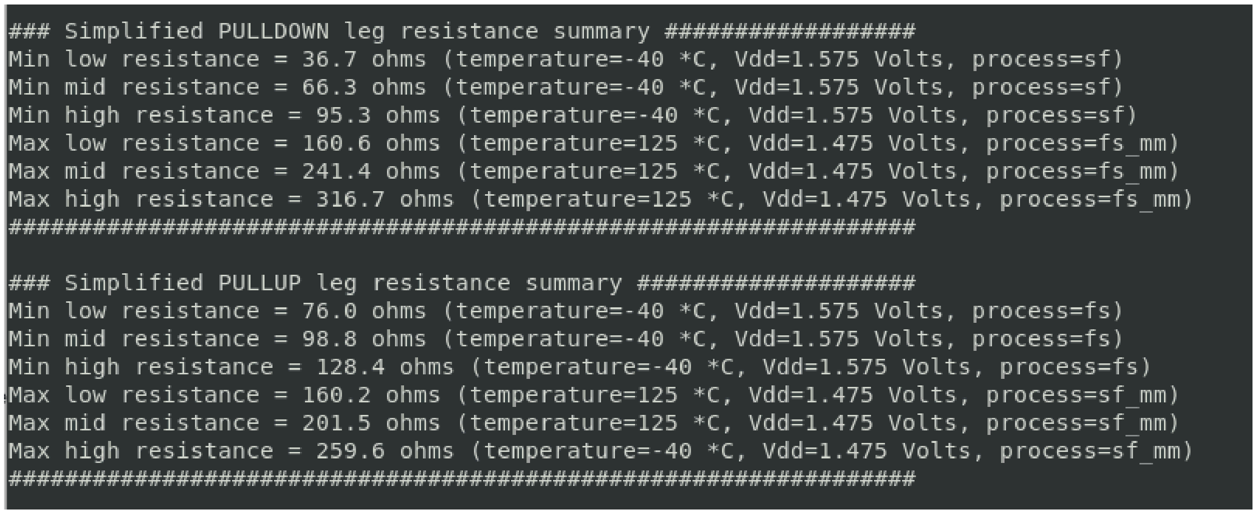
Multiplicity is the number of transistors in parallel, which is in the schematics, considered to be one device.

For the pulldown FETs we used the **normal voltage NFET**, and for the pullup FETs we used the **low voltage PFET**. Even though low voltage PFET has a larger minimum length, we determined it had the best performance for its area. To configure the simulation, set the device properties in `schem/n-leg.sch` , and `schem/p-leg.sch` . Example below:





To measure the max and max resistances at all corners, run the command `make leg-sim`. The top level schematic for the simulation can be found at `schem/n-leg_tb.sch` and `schem/p-leg_tb.sch` . The result of the simulation is shown below:

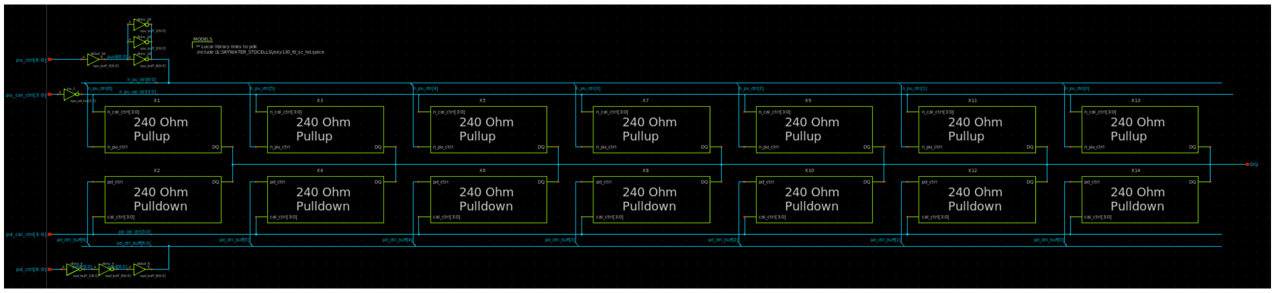


```
### Simplified PULLDOWN leg resistance summary ##################
Min low resistance = 36.7 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=sf)
Min mid resistance = 66.3 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=sf)
Min high resistance = 95.3 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=sf)
Max low resistance = 160.6 ohms (temperature=125 *C, Vdd=1.475 Volts, process=fs_mm)
Max mid resistance = 241.4 ohms (temperature=125 *C, Vdd=1.475 Volts, process=fs_mm)
Max high resistance = 316.7 ohms (temperature=125 *C, Vdd=1.475 Volts, process=fs_mm)
#################################################################

### Simplified PULLUP leg resistance summary ####################
Min low resistance = 76.0 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=fs)
Min mid resistance = 98.8 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=fs)
Min high resistance = 128.4 ohms (temperature=-40 *C, Vdd=1.575 Volts, process=fs)
Max low resistance = 160.2 ohms (temperature=125 *C, Vdd=1.475 Volts, process=sf_mm)
Max mid resistance = 201.5 ohms (temperature=125 *C, Vdd=1.475 Volts, process=sf_mm)
Max high resistance = 259.6 ohms (temperature=-40 *C, Vdd=1.475 Volts, process=sf_mm)
#################################################################
```

For this first simulation, we care only about whether the 'max' resistance requirements are being met. And looking at the chart, we can see this is the case. The closest it comes to failing is the pullup 'Min mid' case where we only meet the 264 Ω requirement by 2 Ω.

Step 5: Combined SSTL simulations

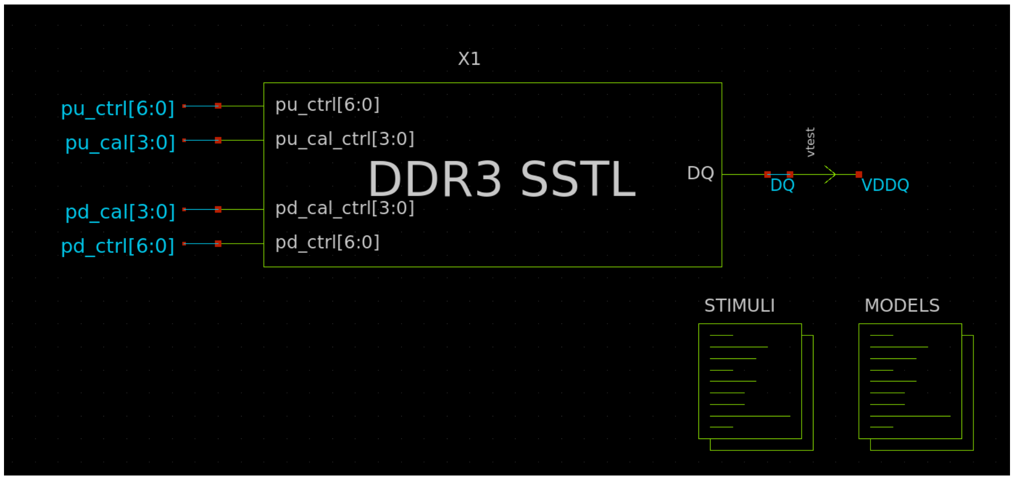
Next, create the top level SSTL circuit. The control signals (enable/disable) for each leg are independent. However the calibration control signals are all connected for both leg types. All DQ signals of every leg are shorted together. The top level schematic file is `schem/SSTL.sch` . Here you can edit the control signal driver circuit as well.



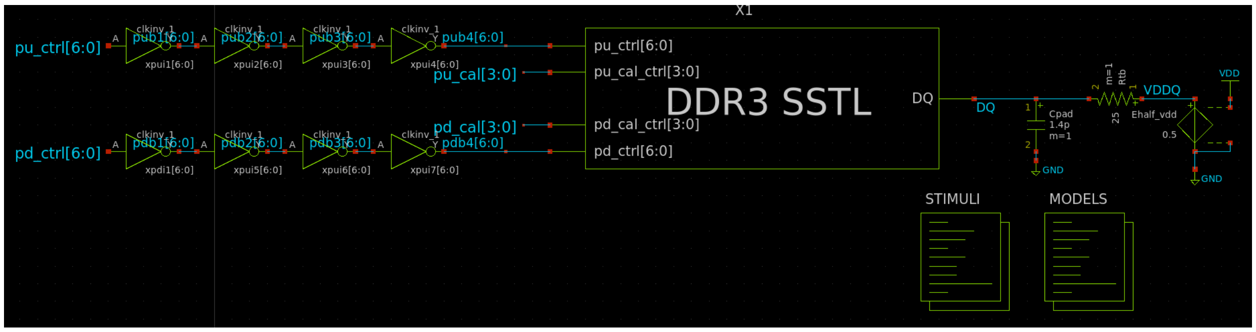
Complete SSTL Circuit

There are two separate simulations needed for the circuit. A DC simulation (to test the resistance requirement) and a transient simulation (to test the slew requirement)

Resistance Simulation (DC)



Slew Simulation (Transient)



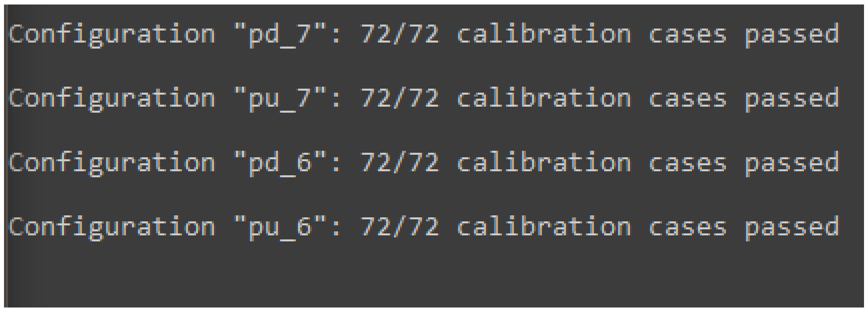
In the above schematic, the chain of inverters for the input signal is just to produce a realistic slew rate. The output load for DQ is to follow the reference load design from the spec:

(JESD79-3E, pg 125.) The capacitor attached to DQ is supposed to estimate the parasitic capacitances to the pad. Parasitic extraction from the circuit layout will result in a much better capacitance estimation.

Results

After completing the SSTL schematic, our next simulation simulates the calibration process at every PVT corner. The calibration process works like a binary search among the calibration FET settings. At the current setting, the resistances are measured. If the resistance requirements are met, the calibration is complete and the valid settings are found. If not, the calibration FET settings are adjusted to increase or decrease resistance. And the simulation is repeated. Since there are 16 calibration options, a calibration will be found in at most 4 tries.

To test calibration at all corners, run the command `make sstl-res-sim`. The top level schematic for this simulation can be found at `schem/sstl_res_tb.sch` . The result of the simulation is shown below.



```
Configuration "pd_7": 72/72 calibration cases passed

Configuration "pu_7": 72/72 calibration cases passed

Configuration "pd_6": 72/72 calibration cases passed

Configuration "pu_6": 72/72 calibration cases passed
```

With these settings, a valid calibration can be found at all corners. For details for each individual simulation, see the files `out/results/sstl_<SIM_SET>_resistance.json` . Where <SIM_SET> is one of pd_7, pu_7, pd_6, pu_6. This refers to the pull-up or pull-down settings of the SSTL, and the 7-leg or 6-leg output resistance.

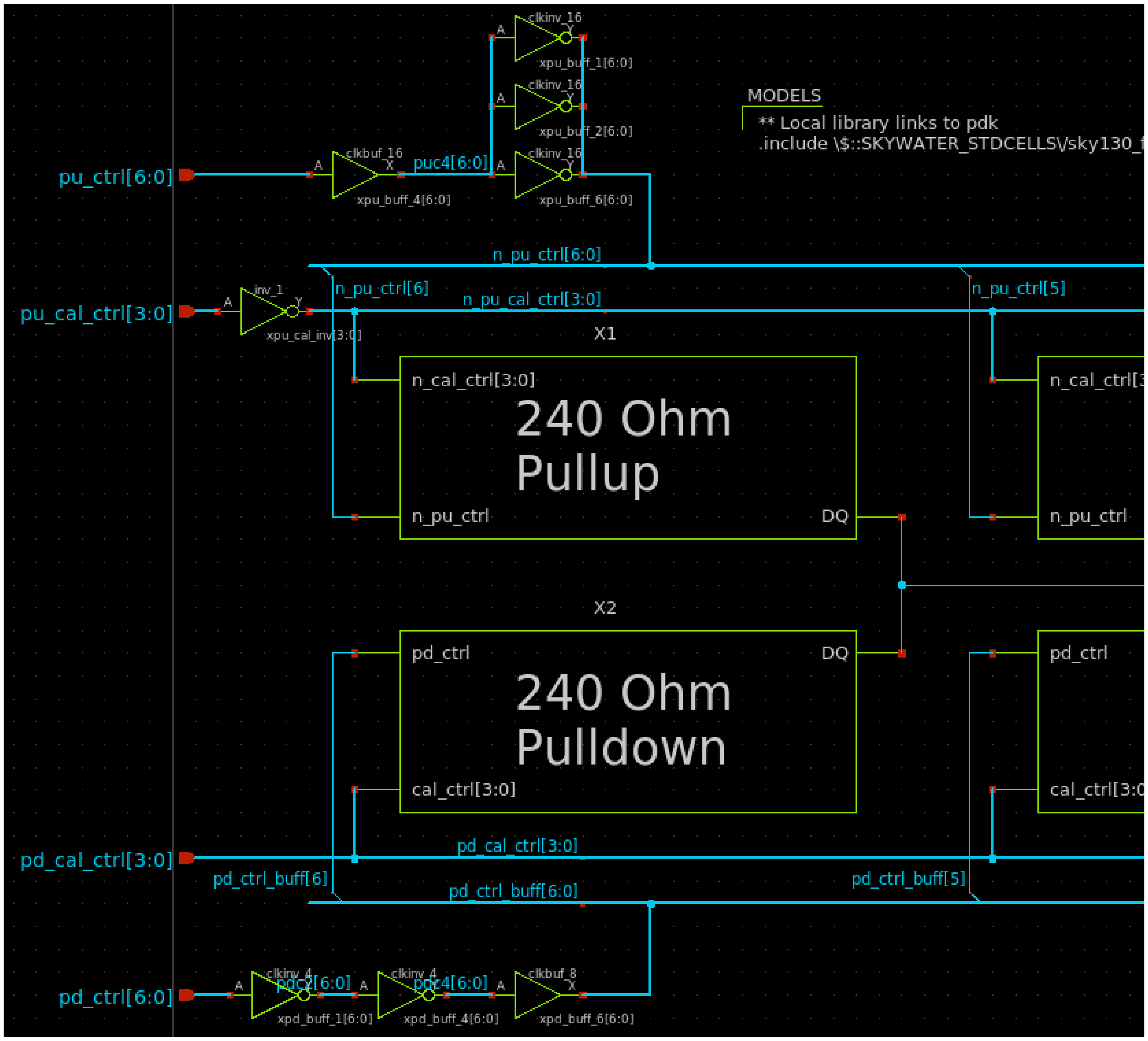
Step 6: Design control signal drivers

With the slew simulation, you can make slew rate measurements at different PVT corners. Slew rate can be controlled by sizing the input buffers driving the leg enable/disable signals.

Set the buffer sizes such that the minimum slew rate is met at the slowest slew rate corner.

Results

After running the simulation, this is the final control signal drivers we decided on:



To test calibration at all corners, run the command `make sstl-slew-sim`. The top level schematic for this simulation can be found at `schem/sstl_slew_tb.sch` . The result of the simulation is shown below.

```
# NOTE: the the spec only specifies a slew requirement for the 7-leg configuraiton,
# so the 6-leg configuration slew tests are not run by default.
python3.6 scripts/sstl_slew_result.py

With 7 legs enabled:
Minimum slew = 3.68 V/ns (slew down, 125*C, 1.575V, "sf" process)
Maximum slew = 7.09 V/ns (slew up, -40*C, 1.575V, "lh" process)
```

Since the requirement is that the slew should always be between 2.5 and 5 V/ns for the fastest DDR3 option, we do not meet that requirement. With the large pull-up FET size, the input capacitance is so high it is difficult to make that leg turn on/off in time.

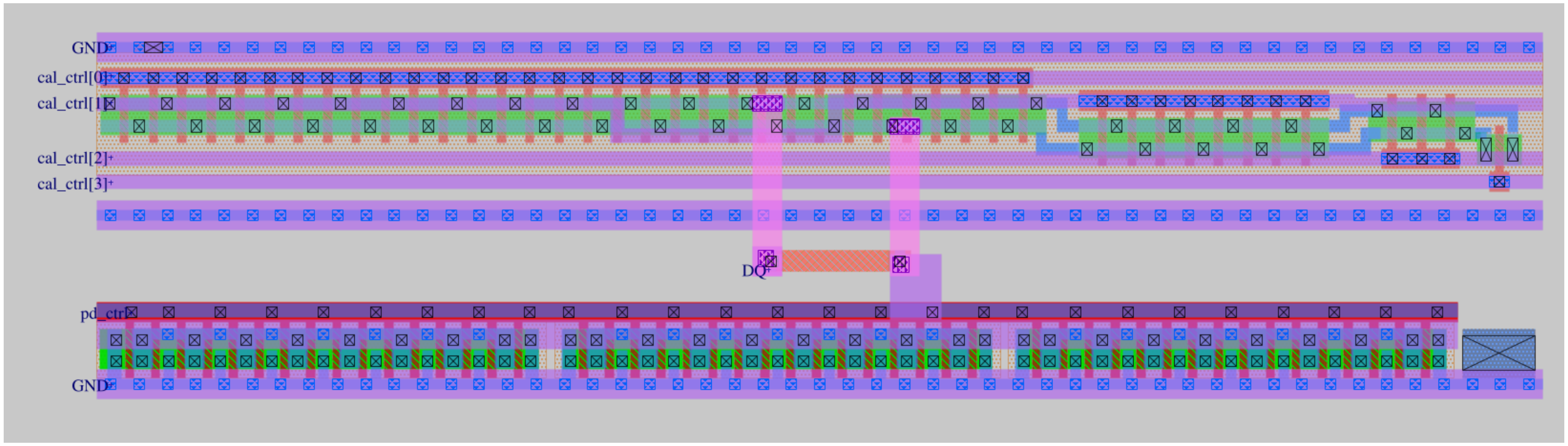
Step 7: Layout

Layout the decided circuit in MAGIC. (See MAGIC instructions below) Complete LVS, then extract a spice model with parasitics (PEX.)

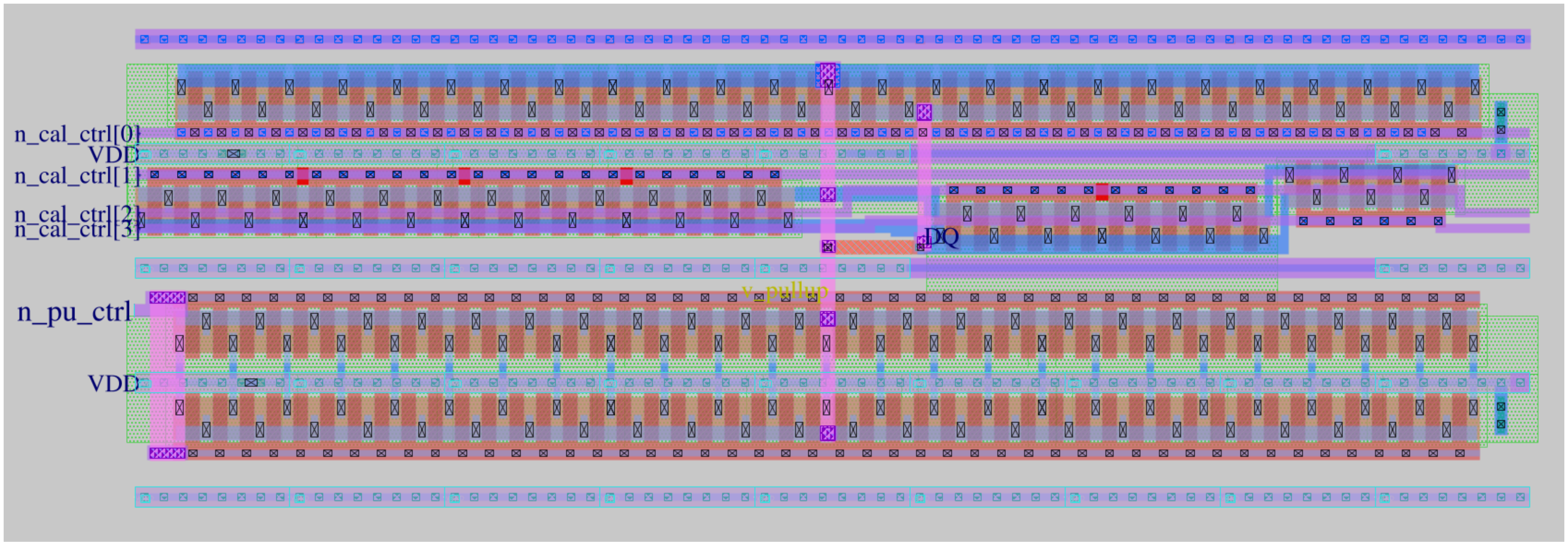
To draw the resistors, use one of the poly-resistor layers. The contacts to other layers are the same as regular polysilicon.

Results

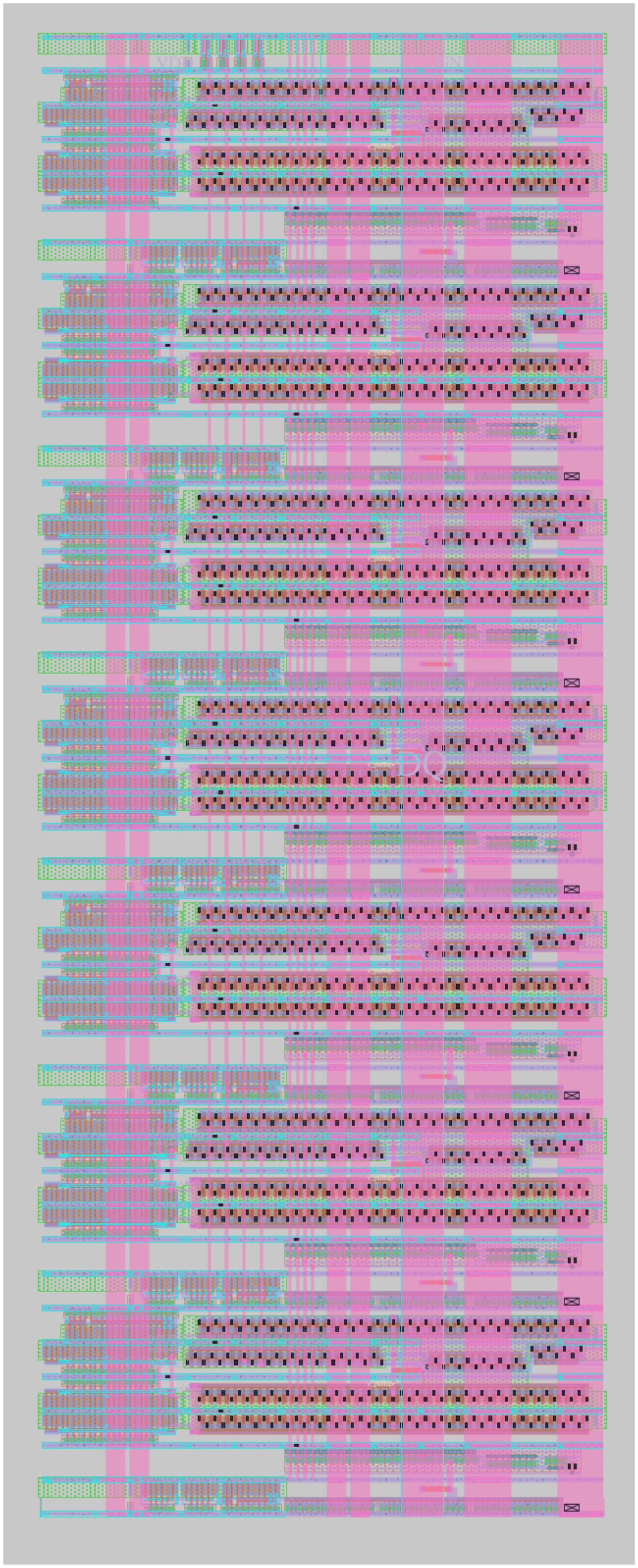
N-Leg layout:



P-Leg layout:



Complete SSTL layout: (Metal layers 3-5 removed)

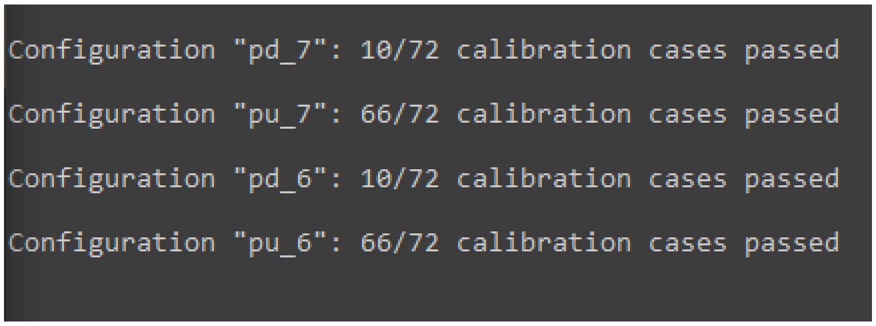


Step 8: Post-layout simulation

Using the same testbenches as before, replace the schematic-generated model of the circuit with the model extracted with parasitics. Run the simulation to see which requirements are still satisfied. If some are not, iterate back to the previous design steps, or alter layout to reduce parasitic components.

Results

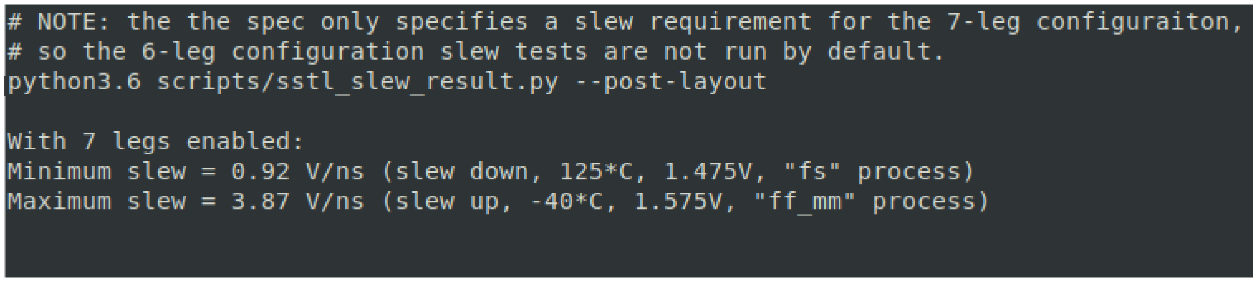
To repeat the SSTL calibration simulation with the post-layout extracted SPICE model, run the command `make post-layout-sstl-res-sim` . The top level schematic for this simulation can be found at `schem/post_layout_sstl_res_tb.sch` .The results for our design are shown below:



```
Configuration "pd_7": 10/72 calibration cases passed

Configuration "pu_7": 66/72 calibration cases passed

Configuration "pd_6": 10/72 calibration cases passed

Configuration "pu_6": 66/72 calibration cases passed
```

With this design, most corners do not have a valid calibration. For details for each individual simulation, and which corners passed or failed, see the files `out/results/sstl_<SIM_SET>_resistance.json` . Where <SIM_SET> is one of pd_7, pu_7, pd_6, pu_6. This refers to the pull-up or pull-down settings of the SSTL, and the 7-leg or 6-leg output resistance.

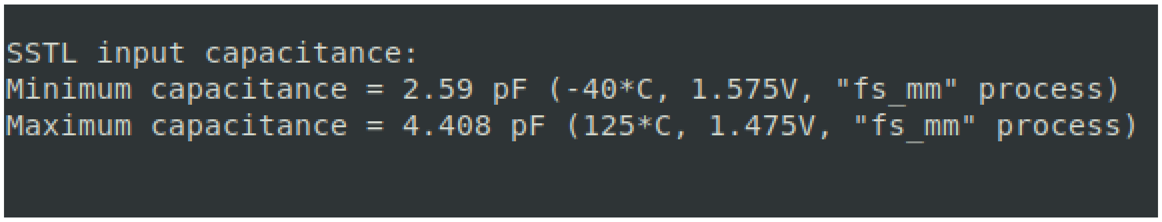
To repeat the SSTL slew simulation with the post-layout extracted SPICE model, run the command `make post-layout-sstl-slew-sim` . The top level schematic for this simulation can be found at `schem/post_layout_sstl_slew_tb.sch` . The results for our design are shown below:



```
# NOTE: the the spec only specifies a slew requirement for the 7-leg configuraiton,
# so the 6-leg configuration slew tests are not run by default.
python3.6 scripts/sstl_slew_result.py --post-layout

With 7 legs enabled:
Minimum slew = 0.92 V/ns (slew down, 125*C, 1.475V, "fs" process)
Maximum slew = 3.87 V/ns (slew up, -40*C, 1.575V, "ff_mm" process)
```

As can be seen from these results the minimum slew requirement (2.5 V/ns) could not always be met. On the slower process corners, the slew is too slow.

One final simulation is the post-layout output capacitance measurement simulation. Run the command make `post-layout-sstl-cap-sim` . The top level schematic for this simulation can be found at `schem/post_layout_sstl_cap_tb.sch` . The results are shown below:



```
SSTL input capacitance:
Minimum capacitance = 2.59 pF (-40*C, 1.575V, "fs_mm" process)
Maximum capacitance = 4.408 pF (125*C, 1.475V, "fs_mm" process)
```

Of the various DDR3 categories, the least strict is DDR3-800, which has a valid capacitance range of 1.4-3.0 pF. Even this requirement is not met, as at the slow corners, the capacitance is too high.

Step 9: Testing

Additionally, we designed a test chip for this circuit, and it was submitted to the Efabless MPW-5 shuttle. (See live Shuttle Status for estimated delivery date.)

We have developed a separate test plan for this chip specified in this doc.

Conclusions

The current PEX extraction model and following SPICE simulations suggest that this circuit is not compliant with DDR3 at some PVT corners.

Particularly the pull-up FETs have high resistance at some corners and this must be compensated for by placing many in parallel. This takes up a lot of area. And since the input capacitance to the main control FET is so high, the drivers for the FET also must have high area. Likely even input buffers will be needed to improve the slew rate. The large area also leads to large input capacitance.

One challenge is the low voltage required for DDR3. The operating voltage for this circuit must be 1.5V to be compliant, but the nominal turn-on voltage for the sky130 FETs is 1.8 V. No lower voltage FETs are available in this PDK.

Additionally, the poly-resistors are likely too small to support the current they are supposed to. According to this obscure page, all poly-resistors only have accurate spice models up to 500 μA/μm. In the low resistance and high voltage case, each resistor will take up to $(1.575 \text{ V})^2$ / (0.9 * 240 Ω) = 11.5 mA. With a width of 330 nm, that is a current density of 34,000 μA/μm. However this should be the worst case, and only would occur momentarily at each transition. I have no idea if this will have a significant impact on the performance or longevity of the circuit.

References:

[1] Micron Technical Note 41-02, “DDR3 ZQ Calibration”, 2008.

[2] F. Plessas, E. Davrazos, A. Alexandropoulos, M. Birbas and J. Kikidis, “A 1 GHz, DDR2/3 SSTL driver with On-Die Termination, strength calibration, and slew rate control”, 2011.

[3] JEDEC Solid State Technology Association, "DDR3 SDRAM Specification, Release 3E", JESD79-3E, July 2010.

Appendix: Getting Started with MAGIC

MAGIC is one the layout tools compatible with the sky130 pdk.

MAGIC documentation main website:
Magic VLSI

Complete MAGIC command list from manual:
Magic-7.4 Command Reference

Helpful video tutorial (although somewhat old):
Magic VLSI Layout Tutorial - part 1

Helpful Shortcuts and Instructions

Shortcut "u" : Short for "undo".

Shortcut "z": Zoom in. "shift-z" to zoom out.

Shortcut "v": Resets zoom to show the entire loaded cell.

Shortcut "x": Short for "expand". Loads the contents of all cells under the rectangle. Subcells are un-expanded by default. "Shift-x" to un-expand cells under the rectangle.

Shortcut "s": Short for "select". Highlights a rectangle of one of the layers under the cursor. If it didn't select the layer you wanted, move the cursor and hit "s" repeatedly until you get the one you want.

Also, if you press "s" repeatedly (without moving the cursor) everything electrically connected to the selected layer will be highlighted. This is very useful to make sure you made all of the connections you intended.

Shortcut "a": Short for "area". Selects all visible layers under the rectangle. (Also selects un-expanded sub-cells for some reason.)

Shortcut "i": Short for "instance" (I think). Selects a subcell instance under the cursor. If it didn't select the subcell you want, press "i" repeatedly until you get the one you do.

Shortcut "," (comma): Unselect whatever is currently selected.

Shortcut "m" : Short for "move". Move whatever is currently selected to the cursor position. Note, the selection is moved relative to the lower left corner of the current rectangle.

Shortcut "c" : Short for "copy". Copy whatever is currently selected to the cursor position. Note, the selection is copied relative to the lower left corner of the current rectangle.

Shortcut "d" : Short for "delete". Removes what is currently selected.

Shortcut “>” : Descents (or “pushes”) into the selected sub-cell. So you are now editing the selected cell.

Shortcut “<” : Ascends (or “pops”) up into the parent cell. So you are now editing the parent of the previously loaded cell.

Instruction “select area <layer>”: Identical to the “a” shortcut, except only selects rectangles of the requested layer.

Instruction “select area <less/more> <metal-layer>”: Same as previous instruction, but subtracts or adds to the current selection. This is helpful to select multiple metal layers under the current rectangle.

Instruction “label <label-name>”: Create a label of the given name at the current rectangle. Make sure the label went to the layer you wanted. This can be checked and set with the “setlabel command”

Instruction “setlabel <option> <value>: Edits the properties of the selected label(s). This command with no arguments prints a list of the options. This command with no <value> field prints the current value of the option of the selected label(s).