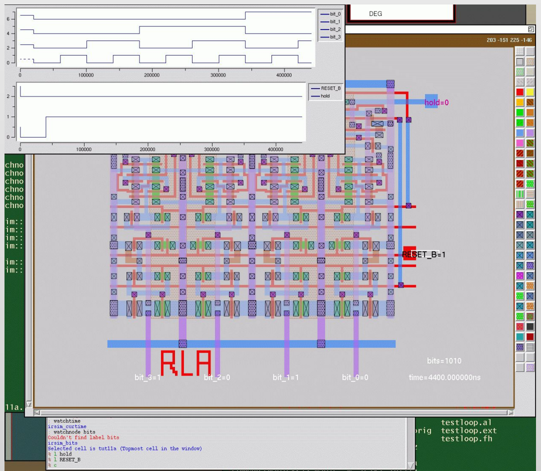
IRSIM: A Switch-Level Simulator and Dynamic Power Analysis Tool

Jason Liang, Timothy Edwards

3 November 2022

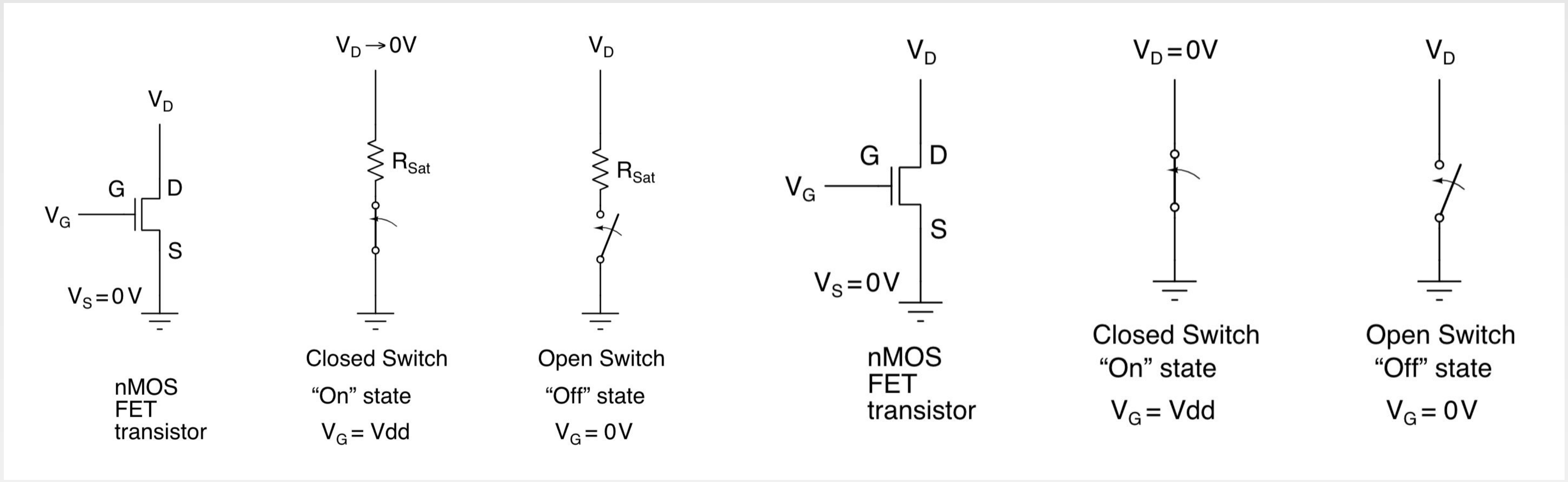


```
: watchtime
irsim_curtime
: watchnode bits
Couldn't find label bits
irsim_bits
Selected cell is tut11a (Topmost cell in the window)
% l hold
% l RESET_B
% c
```

Introduction

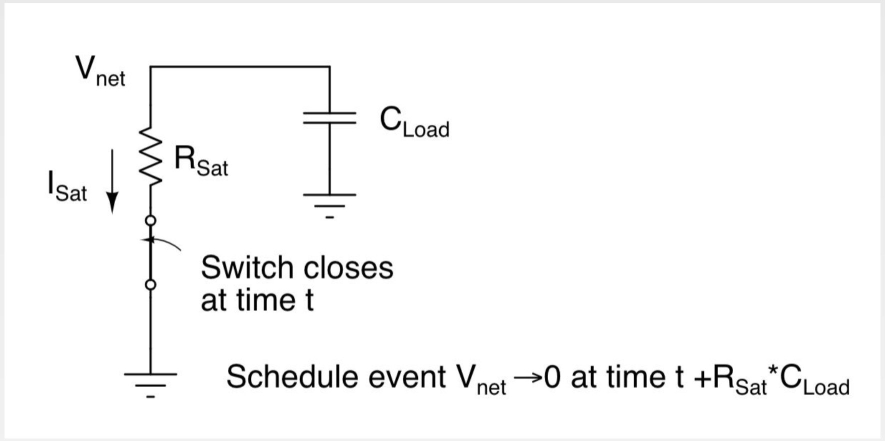
- Originally developed for fast logic simulations before HDLs
- Two types of transistor models for simulation: switch and linear
- Includes a less advertised dynamic power analysis tool
- One of few open source tools that can do dynamic power analysis
  - OpenSTA (power analysis using .lib files)

LINEAR **SWITCH**

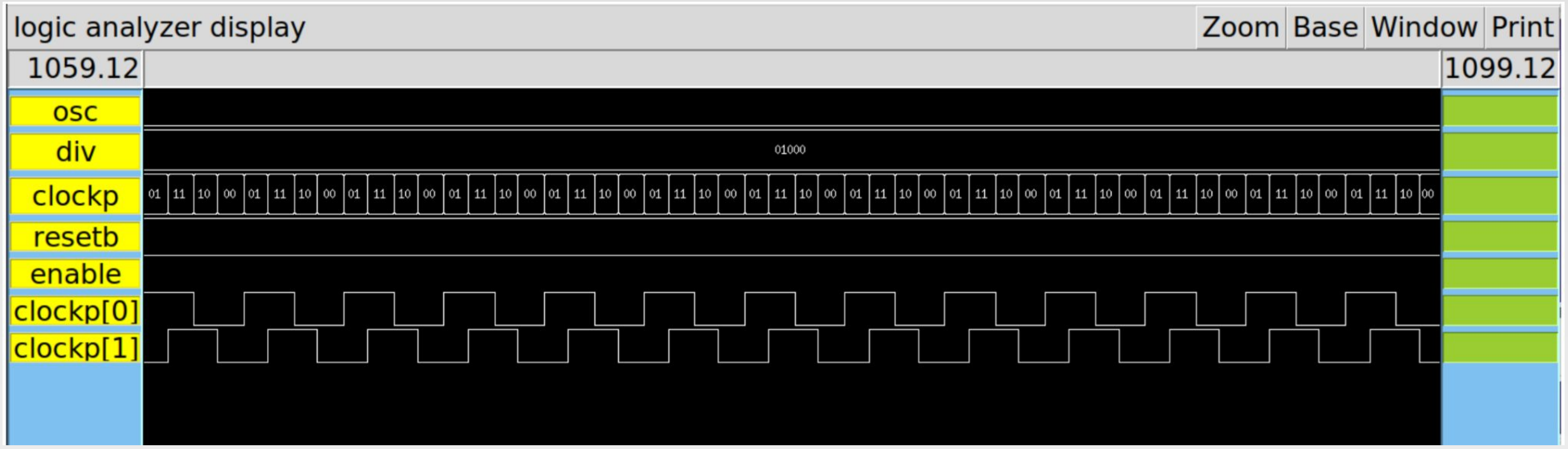


Simulation

- Supports Verilog-style testbench commands for verification and simulation of circuit (`at` and `every`)
- Simulation by scheduling events
  - User feeds stimuli into interpreter
  - Each stimulus provided is an event to be scheduled
  - If a circuit is modified and resimulated, only the parts that have been changed are resimulated
- Events are timed approximately using an RC constant


V_{net}
C_{Load}
R_{Sat}
I_{Sat}
Switch closes at time t
Schedule event $V_{net} \rightarrow 0$ at time t $+R_{Sat}*C_{Load}$

Simulation


logic analyzer display
Zoom
Base
Window
Print
1059.12
1099.12
osc
div
clockp
resetb
enable
clockp[0]
clockp[1]
01000
01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00 01 11 10 00

Simulation

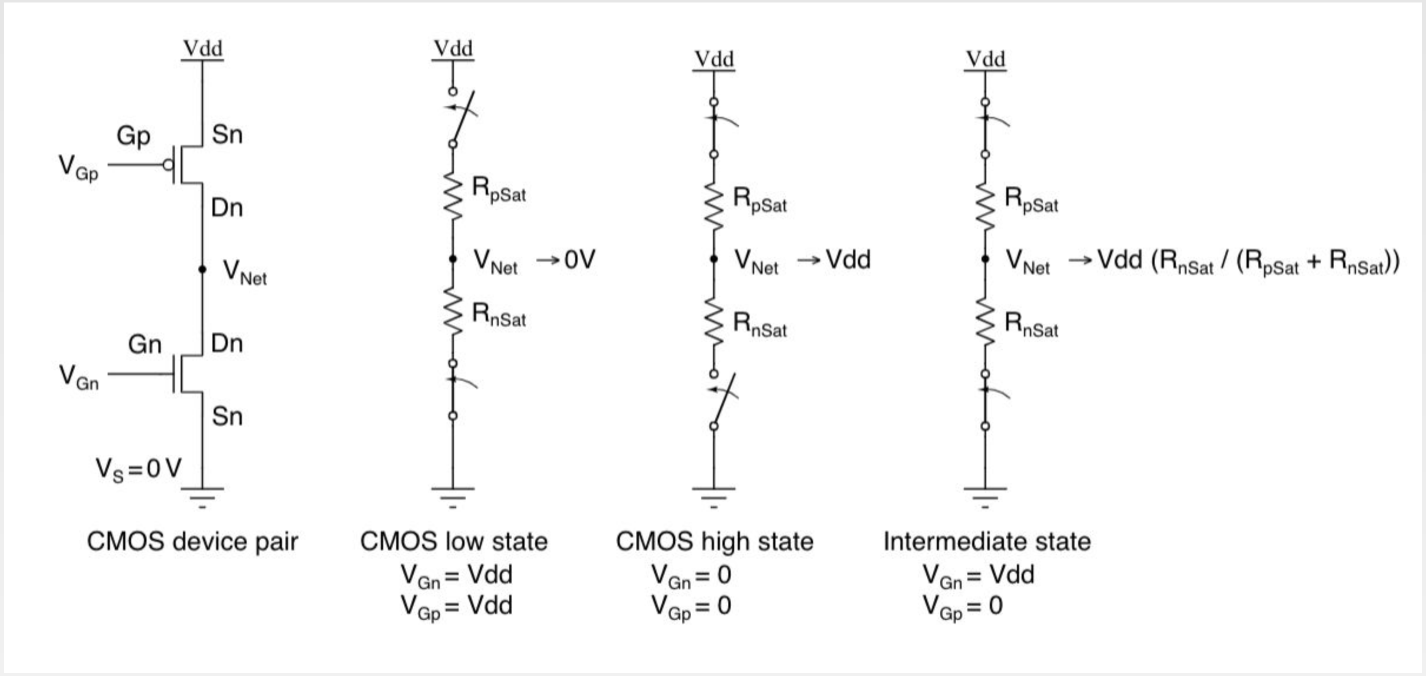
- Event constitutes switching of transistor state, similar to state changes in Verilog simulations
- Performs orders of magnitudes faster than SPICE simulations but still slower than Verilog
- SPICE provides power analysis information but Verilog does not, users simply stuck to the main purposes of other tools and ignored IRSIM

Icarus Verilog

NGSPICE

Dynamic Power Analysis

- Functioned as originally intended but lacked useful features
- Dynamic power is normally calculated using the equation at right
- IRSIM calculates switching energy of every transistor charging or discharging into capacitive load



$$P = \frac{CV^2}{2t}$$

IRSIM Before and After

BEFORE	AFTER
Supported only one nFET and one pFET device type	Supports an arbitrary number of FET device types
Supported simulation under a single power supply	Supports simulation under multiple power supply
Needed to have commands for verifying power behavior	Supports commands for power visualization over given simulation time

UCSD Jacobs School of Engineering

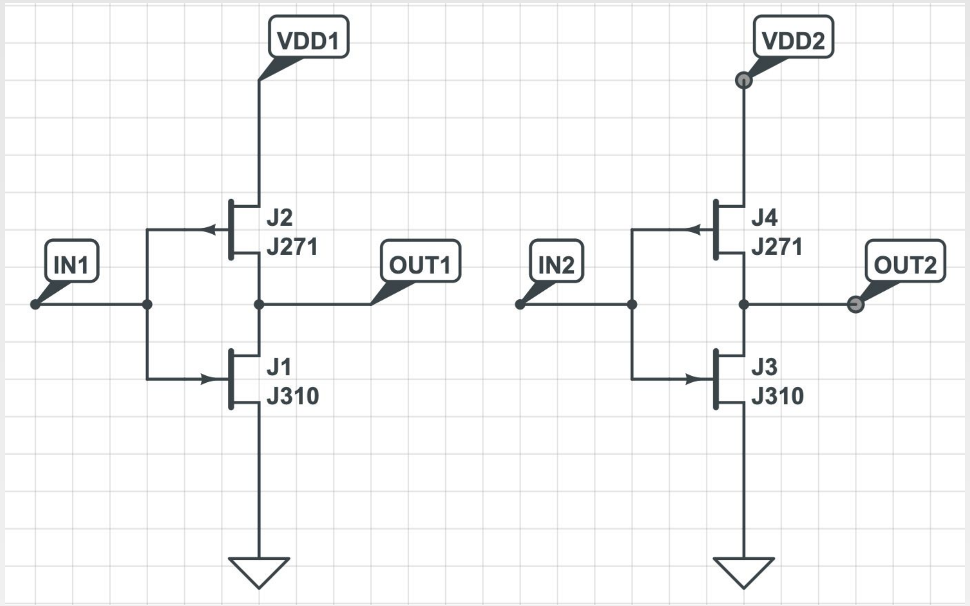
Multiple Device Support

- ".sim" file is netlist of device characteristics in circuit, typically generated from layout tool
- Contains 'x' device type for subcircuit defining arbitrary circuit component beyond transistor/resistors/capacitors
- Updated to include device keyword for mapping foundry device models to IRSIM pFET and nFET devices:
  - `device nfet|pfet <device name>`
- Specify custom resistance types
  - `resistance <device name> static|dynamic-high|dynamic-low <width> <length> <resistance>`
- Device parameter files after this update: https://github.com/RTimothyEdwards/open_pdks/tree/master/sky130/irsim

UCSD Jacobs School of Engineering

Multiple Power Domain Support

- **Effective for dual-voltage domains with thick and thin oxide devices or standby power domains**
- **Specify a second power source using `vsupply`**
  - `vsupply <power name> <power>`
- **After each step time, total energy is accumulated instead of total capacitance: $\Sigma(CV^2)$, saves computation time**



Multiple Power Domain Support

- **Example commands to support this:**

```
# declare all of the power node names
power <net name>
...

# declare the ground node
ground <ground name>

# initialize logic high nodes and logic low nodes
h <node>
...
l <node>
...

# optionally, toggle signals
every <time> {toggle net}

# set voltage values
vsupply <power net> <voltage value>
```

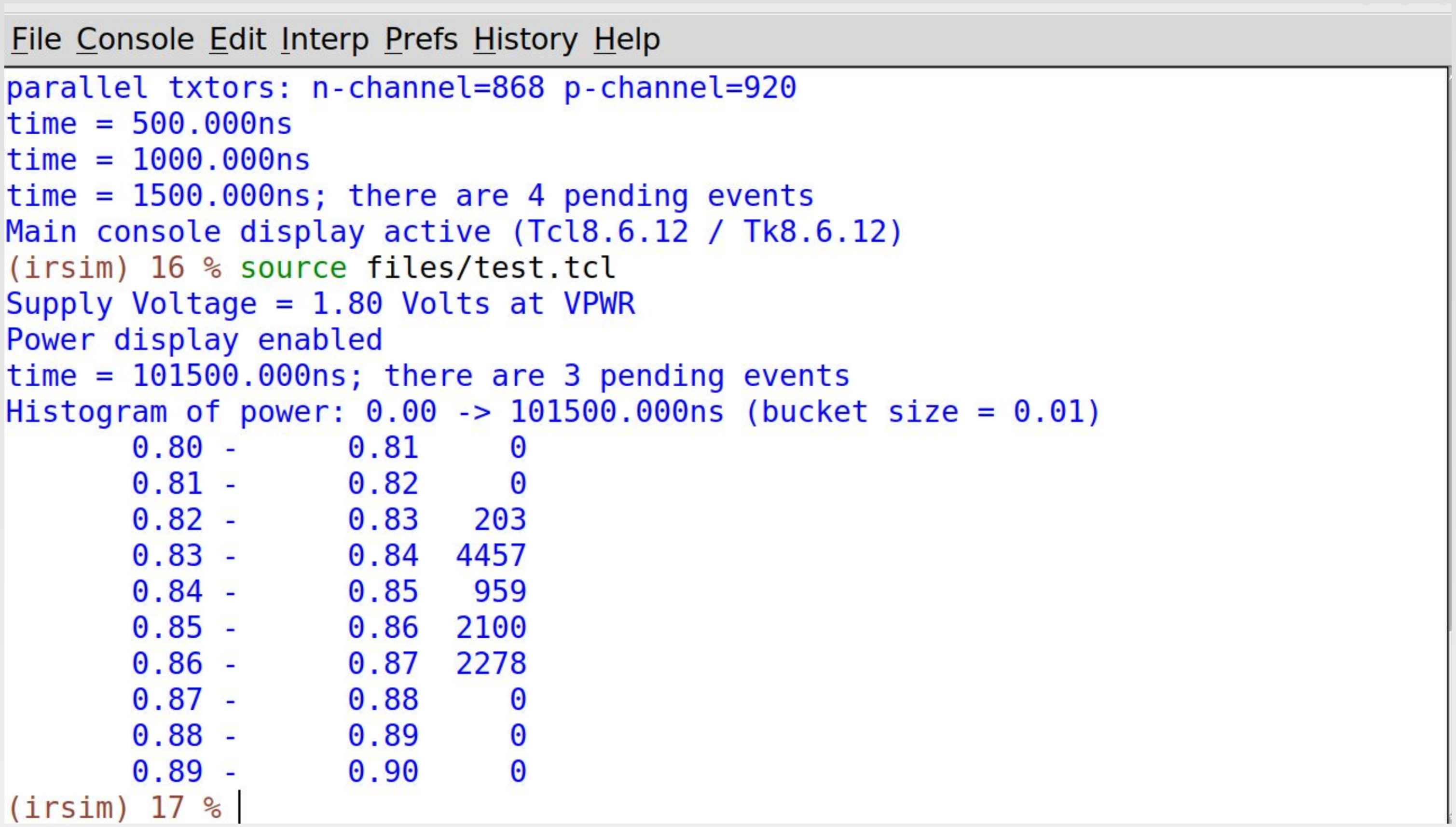
Histogramming

- **Power visualization tool that allows user to visualize power behavior while simulating circuit**

```
# set up circuit (voltage, power supply, etc…)
powlogfile /dev/null
powtrace *
powstep

# initialize the histogram with the minimum and maximum power range
# and optionally the number of buckets, 20 default
powhist init [min] [max] <buckets>

# if the histogram structure has been initialized, use the power
# estimation formula to record the power and store it in the
# histogram, otherwise, throw an error
every <n> {powhist capture}

# simulate circuit indicated by time argument, capture power every n
nanoseconds
s <time>

# clear the histogram data and free the memory
powhist reset

# print captured histogram data stored
powhist print
```

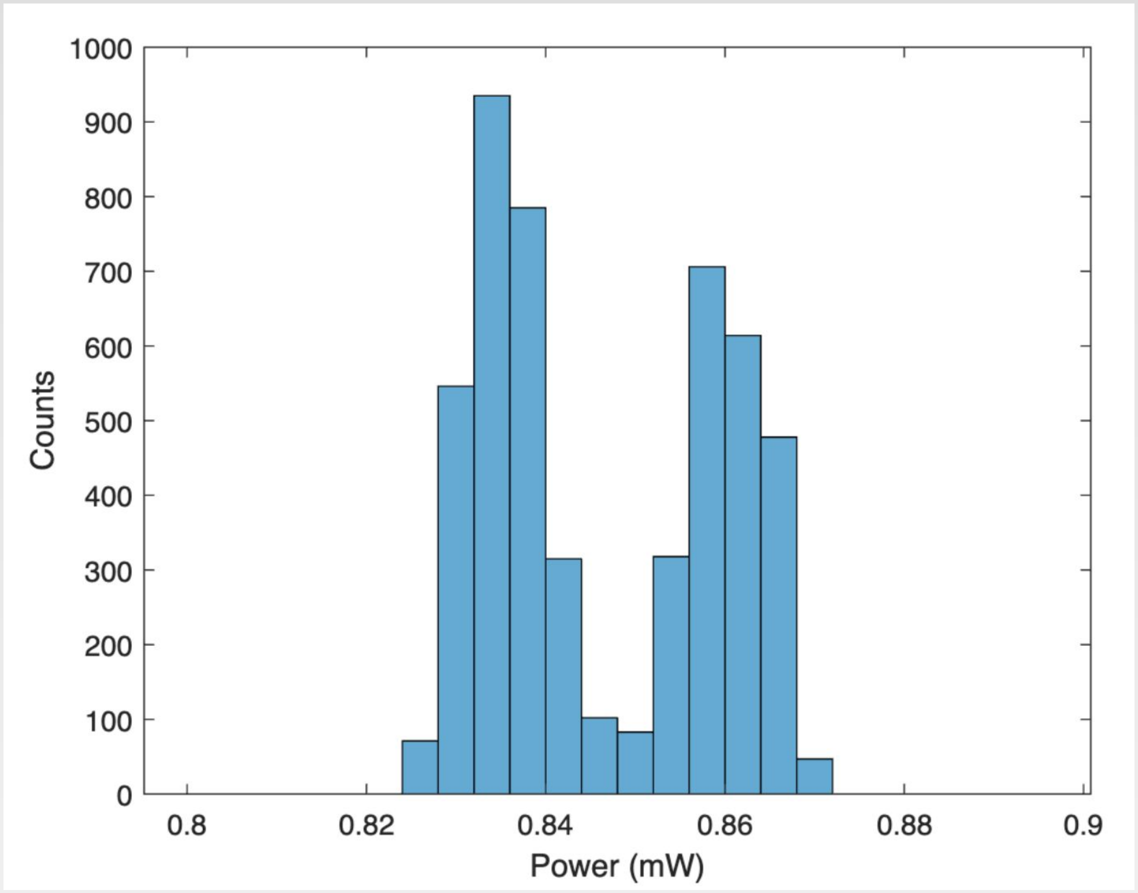
UCSD Jacobs School of Engineering

Histogramming



```
parallel txtors: n-channel=868 p-channel=920
time = 500.000ns
time = 1000.000ns
time = 1500.000ns; there are 4 pending events
Main console display active (Tcl8.6.12 / Tk8.6.12)
(irsim) 16 % source files/test.tcl
Supply Voltage = 1.80 Volts at VPWR
Power display enabled
time = 101500.000ns; there are 3 pending events
Histogram of power: 0.00 -> 101500.000ns (bucket size = 0.01)
       0.80 -      0.81     0
       0.81 -      0.82     0
       0.82 -      0.83   203
       0.83 -      0.84  4457
       0.84 -      0.85   959
       0.85 -      0.86  2100
       0.86 -      0.87  2278
       0.87 -      0.88     0
       0.88 -      0.89     0
       0.89 -      0.90     0
(irsim) 17 %
```

Histogramming



Conclusion

- Three major updates to IRSIM:
  - multiple nFET/pFET device support
  - multiple power voltage domain support
  - power visualization command-line package
- Future work includes modifying delay and voltage compute model to accommodate parasitic interconnect delay (file format parsing)
- Repository link: https://github.com/RTimothyEdwards/irsim
- Acknowledgements: Dr. Tim Edwards and the Google Summer of Code program for facilitating and mentorship

UCSD Jacobs School of Engineering