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

Jason Liang
Electrical and Computer Engineering Department
UC San Diego
San Diego, California
jsliang@ucsd.edu

R. Timothy Edwards
SVP Analog and Platform
Efabless, Inc.
Palo Alto, California
ORCID 0000-0001-8772-0753

***Abstract*—IRSIM is an open-source switch level simulator that analyzes digital circuits at the device (transistor) level using a linear switch-based model that depends on whether a transistor is "on" or "off" but accounting for the resistance through the device for delay estimates. This event-based simulation makes IRSIM much faster than a device modeling simulator like SPICE, but slower than a Verilog simulator, which models circuits at the gate level yet lacks a power analysis component. However, since Verilog simulations are still much faster than IRSIM for logic simulations, IRSIM has been largely ignored, and its dynamic power analysis feature forgotten. In this paper, we discuss the software methods in IRSIM with particular emphasis on further improving its dynamic power analysis capabilities.**

***Index Terms*—transistor, circuit simulation, timing, switching, dynamic power**

I. Introduction

Power consumption is an increasing concern in today's high performance digital designs. Due to advances in VLSI manufacturing technologies resulting in increasing device densities and performances, power efficiency is becoming more important than ever. Proper dynamic power analysis requires calculating switching power while simulating a device undergoing switching activity as clocks are toggled between high and low logic values and parts of the circuit are put into specific operating modes. Although originally developed to address the need for fast and reasonably accurate digital circuit simulation prior to the development of Verilog and VHDL, IRSIM has also been given a set of power estimation commands, though it is a less advertised feature. Its main functionality is incremental simulation [1] in which each transistor in a design is modeled as a voltage-controlled resistive switch. This simplified linear model performs quite well; each FET transistor can be in either an ON or OFF state and events are added to a queue to be evaluated each time a transistor switches states. Turning a transistor ON means replacing the switch with a resistor connecting the source and drain nodes. The voltage through a resistor can then be one of three values: logic HIGH, logic LOW, or undetermined, and is determined by solving for the voltage in a resistor network formed by the current path of ON transistors. Users can also modify the circuit for incremental resimulation, in which only parts that have been modified need to reevaluated. IRSIM contains a digital simulation analyzer that allows for debugging circuit behavior, similar to Verilog testbenches, as well as a Tcl-command line interface for controlling these simulations, generating custom user subcircuits, and performing power analysis.

II. Background

A. Simulation by Events

Simulation in IRSIM is initiated by parsing input stimuli from a file or the command line, which are interpreted as events to be scheduled in a time queue. As simulation advances, events that are scheduled to occur are removed from the queue and all the node voltages specific to these events change values. If these nodes are part of current paths of ON transistors, the node voltages along these paths are also marked for reevaluation. If any calculations change from the current value, an event is scheduled for transition, whose occurrence is scheduled by using an RC time constant delay. This process is repeated until no more events are left in the queue at the end of the simulation time.

B. Power Analysis

Many techniques and tools exist for computing the power consumed in digital and mixed-signal circuits, but few open source tools exist for these purposes. The OpenROAD project [7], an open source RTL to GDS generation workflow, uses a static timing analyzer (STA) tool called OpenSTA for instance-based power reporting. STA tools perform power analysis similar to timing analysis by reading in power parameters in the associated standard cells indicated in Liberty (.lib) files. Dynamic power is then calculated by summing the switching power and the capacitance load power and multiplying by a constant representing the average fraction of transistors switching on any given clock cycle, yielding a single value. This method only estimates average dynamic power usage and so cannot analyze situations such as power mitigation techniques, where parts of a circuit are placed in a low-power mode, a feature that has recently seen extensive development.

Typically, dynamic power is estimated using the formula

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

where C is the switching capacitance, V is the voltage, and t is the frequency, typically a clock period. The numerator

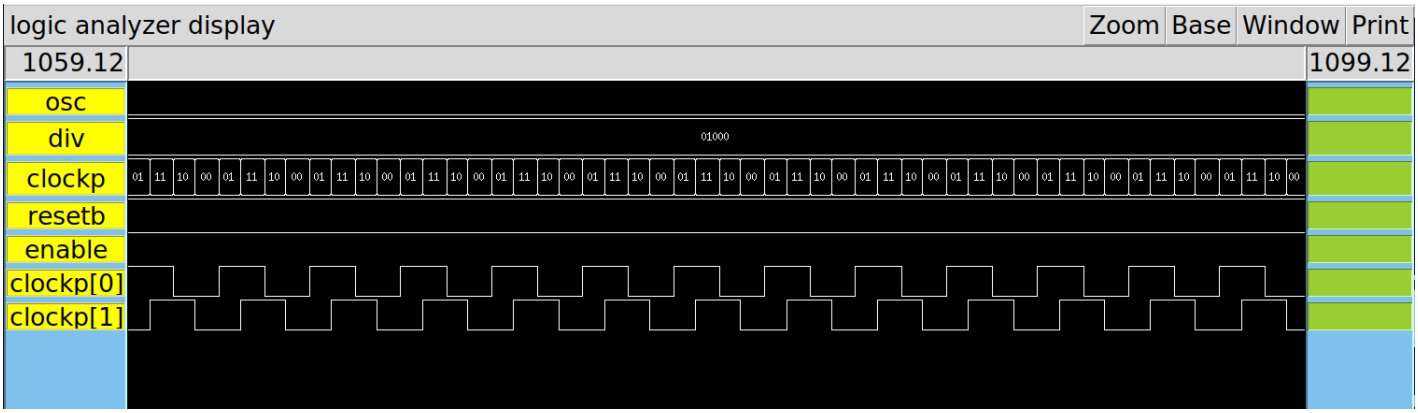

Fig. 1. Logic analyzer simulation for digital ring oscillator

scales with the switching activity, so the more frequently a voltage value changes logic levels, the greater the power dissipation. By contrast, IRSIM calculates the switching energy of every transistor charging or discharging into a capacitive load. Then, as simulation time progresses, switching energy is accumulated and power is computed at the end as the total energy over the time period simulated.

III. Dynamic Power Analysis

A. Existing Features

For digital simulation, users provide a parameter (".prm") file that specifies the process technology the circuit is to be manufactured in and any other process-dependent data, as well as a ".sim" file for the circuit netlist description. Users then provide input stimuli in the form of Tcl commands, which may be in a script, specifying events that are to be entered and scheduled in a time queue. IRSIM features an "analyzer" (GUI) display, and signals to watch can be specified using the keyword `ana`, followed by a vector of signals [3].

Power analysis code in IRSIM prior to the current development work allows users to load in a log file for writing nodal transitions. They can then set the specific nodes they want to trace and the power supply voltage, and then simulate the design for a specified amount of time. For example, we consider a digital locked loop circuit on the open source Caravel SoC [2], which contains a digital controller and ring oscillator for high speed clock generation. The simulation GUI outputs the waveforms shown in Fig. 1. A generic power analysis script generally consists of the following commands:

```
powlogfile <out file>
powtrace <nodes to watch>
powstep
vsupply <voltage>
s <time>
sumcap
```

While this code works and produces a power calculation result, it is minimally useful to a designer and has a number of limitations. The remaining sections describe the development work done to enhance IRSIM's dynamic power analysis.

B. Handling Multiple Device Types and Parameters

In IRSIM code prior to this development work, IRSIM only supported characteristics of a single pair of transistor devices, one nFET and one pFET. The authors extended the IRSIM code to support circuits with multiple device types [6]. While the ".sim" file format used by IRSIM represents all transistors by either 'n' or 'p' tokens, there is also an 'x' token, indicating the use of a subcircuit. While the subcircuit token was originally used as a special simulation type in IRSIM, it is also convenient to use to describe transistors and other devices, where it can overcome the limitation of the original 'n' and 'p' models, and represent transistors of different types and with different characteristics (i.e., on and off resistances). To this purpose, the '.prm' input format used by IRSM was extended to add a `device` keyword that allows a name of a foundry device model to be mapped to an IRSIM nFET or pFET simulation device, with the simple syntax:

```
device nfet|pfet <device name>
```

To specify the three resistances used by IRSIM to model a transistor (`static`, `dynamic-low`, and `dynamic-high`), the original `resistance` was modified to take a device name as the second argument, allowing every device type to be given its own unique characteristics:

```
resistance <device name>
    static|dynamic-high|dynamic-low
    <width> <length> <resistance>
```

Accompanying this format change, the layout tool Magic [4] was modified to produce the 'x' record output for .sim files, and IRSIM support for the SkyWater sky130 process node in the `open_pdks` tool [5] was modified to generate the new format .prm files. The new extended format maintains backward compatibility with all legacy parameter and simulation files.

C. Handling Multiple Power Domains

By being able to analyze circuits with multiple device types, IRSIM can effectively analyze circuits with multiple power

domains, either dual-voltage domains with thick- and thin-oxide devices, or use of a standby power domain, which is common in low power design methodology: In order to reduce leakage power, some transistors are put in a "sleep mode" at a lower voltage and isolated from the primary power supply, also known as power gating. To support this functionality, the existing `vsupply` Tcl command was modified to take a second power node parameter and its own voltage value to specify multiple voltage domains. The `power` command (used to declare a power supply net) functionality was also modified to be able to be called multiple times, indicating that the circuit contains multiple nodes connected to different power supplies instead of the program assuming a single global power supply. To facilitate computing power during simulation time, a routine was developed to search for all transistors connected to a given power supply and then follow each transistor fanout and set the power supply it connects to. The power calculation was also modified to accumulate total energy instead of capacitance after each step (defined by the simulation time) using the formula

$$E = \Sigma(CV^2)$$

where C is the capacitance and V is the voltage supply of a node. This method allows the multiple voltage domains to be handled with very little additional overhead compared to the original power calculation method.

D. Histogramming

Sometimes when simulating a design, a user may want to record the power at meaningful time intervals, such as the clock period of the design, so they can set up testbenches that capture accurate per-cycle average, peak, and leakage power in order to verify the behavior of the circuit. Power calculation from static timing analysis does not provide any detailed information as to how a circuit behaves cycle to cycle, as it only provides a single power estimate. A histogram feature was developed for IRSIM to address this concern, which displays a range of power measurement counts gathered over simulation time. This effectively captures the power distribution of the circuit, allowing novel insights into circuit activity. For instance, circuit designs whose power distributions exhibit a Gaussian distribution indicate that switching transistors have an approximately even distribution from cycle to cycle, while a bimodal power distribution may indicate a switching activity between two different activity domains. The tuned ring oscillator circuit (digital locked loop) in Figure 1 experiences an oscillation between a lower and a higher power state as shown in Fig. 2, which is a histogram measuring power every 10 ns for a total simulation time 50,000 ns. The clearly bimodal distribution represents the system attempting to keep the oscillator tuned by removing a stage from the ring oscillator to speed it up (lower power state) or adding a stage to the ring oscillator to slow it down (higher power state). This sort of analysis can only be made with cycle-accurate dynamic power simulation.

The following Tcl script is used to generate histogram data:

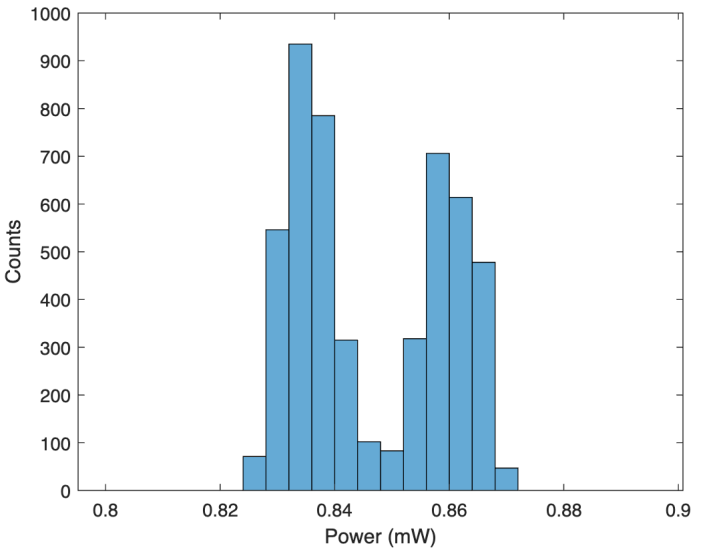

Fig. 2. Power histogram for digital ring oscillator

```
power <power/ground node(s)>
h <power node(s)>
l <ground node>
vsupply <power node(s)> <voltage>
powlogfile /dev/null
powtrace *
powstep
powhist init <min> <max> <bins>
every <timestep> {powhist capture}
s <simulation time>
powhist print
```

IV. Conclusion

In this paper, we introduced the need for power analysis tools for low-power chip design and discussed an open-source solution and its features, both as a dynamic power analysis tool and also as a digital logic simulator. In particular for the former, we introduce various command and file format additions and modifications that allow users to visualize switching power activity via histogramming, use input device parameter files with more transistor types, and simulate circuits with more than one power supply.

Future steps include modifying the delay and voltage compute model to accommodate parasitic interconnect delays (as from a SPEF for SDF format file), a relatively straightforward addition to the circuit models in IRSIM and mostly a format parsing problem.

References

[1] A. Salz and M. Horowitz, "IRSIM: An Incremental MOS Switch-Level Simulator," 26th ACM/IEEE Design Automation Conference, 1989, pp. 173-178, https://doi.org/10.1145/74382.74412.

[2] Efabless. Caravel, 2021. URL https://github.com/efabless/caravel

[3] Open Circuit Design. IRSIM, 1989. URL http://opencircuitdesign.com/irsim/

[4] Open Circuit Design. Magic, 1983. URL http://www.opencircuitdesign.com/magic/
[5] Open Circuit Design. Open_PDKs, 2020. URL http://opencircuitdesign.com/open_pdks/
[6] R. Timothy Edwards. IRSIM, 2002. URL https://github.com/RTimothyEdwards/irsim
[7] UC San Diego VLSI CAD Laboratory. The OpenROAD Project, 2018. URL https://theopenroadproject.org/