2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

NVVM: A Netlist-based Verilog Verification Methodology for Mixed-Signal Design

Jiping Qiu, Texas Instruments, MCU NVM, Dallas TX, USA, j-qiu@ti.com

Kurt Schwartz, Texas Instruments, MCU NVM, Dallas TX, USA, k-schwartz@ti.com

***Abstract*—In order to verify a large mixed-signal circuit, we present a new verification flow that can balance the accuracy and speed. A Verilog netlist is generated from the full-custom schematic, and with appropriate Pre/Post-processing, it can conduct fast function verification in the event-driven simulator. The netlist-based model is able to find the bugs in the real circuit design, as opposed to the spec-based behavioral model, and it is also hundreds of times faster than the low-level Spice model.**

Keywords—Verilog; Netlist-based model; Real-number model; Mixed-signal verification

I. INTRODUCTION

Mix-signal SOC has become commonplace. At an advanced technology node, it is possible to put analog circuits and digital gates together inside a chip, delivering low-cost integrated solutions for automotive, wireless, security and many other applications. The SOC also ends up much more complex, though. Not only does the number of transistors increases, but the analog content of SOC are not pure analog anymore. For the purpose of IP reuse, digital logics are always included in a mixed-signal block, supporting a large number of settings for a wide spectrum of functional and environmental requirements.

How to verify these complex mixed-signal designs is a rising issue. In industry, most SOC design re-spins are due to mixed-signal errors, and each time it costs an extra 5 to 10 million dollars and an 8 to 10 week delay [1], but there is no robust methodology for this kind of verification. The traditional SPICE simulation and black-box digital model are still commonly used.

The FastSPICE approach partitions a large design into smaller sub-circuits that can be solved in different modes and abstract levels, and with some speed-accuracy trade-off, it is possible to simulate a multi-million gate IP. However, detailed verification is only possible at block level. At IP level, when the interaction between blocks are taken into consideration, even the nominal and the worst-case simulation may cost days or weeks, let alone full functional coverage.

Since the SPICE simulation is a bottleneck, black-box digital models of such IPs are then provided to SOC. The over-simplified model is usually extracted from the original specification, and is irrelevant to the actual analog implementation. It may find out some timing violation of the external signals from the rest of SOC, but it is unable to verify any internal functionalities. In modern mix-signal IP there can be hundreds of power modes, operating modes, test modes and trim settings, and leaving them all untested can be very dangerous. In the absence of simple checks, many mixed-signal designs are affected by such avoidable and “highly embarrassing” bugs [1] as logic faults, connection errors and incorrect bus orders.

To strike a balance between the speed and accuracy of full chip simulation, many mixed-signal engineers turn to analog behavioral modeling written in Verilog-A. It uses a time-driven analog kernel to continuously solve the Kirchhoff's equations like SPICE, but the amount of details may vary depending on the desired abstraction level. Although this approach can provide a 5-100X speedup over SPICE, it all depends on how good the modeling is. The modeling person needs to be familiar with the analog circuits to retain exactly enough details, and he also needs to be familiar with programming and debugging the HDL language. The advantage in runtime can be easily offset by the great effort to create Verilog-A models.

Another practice, which is promoted by Cadence, is RNM (Real Number Model). The RNM technique uses discrete real numbers to present voltage and current, and defines a block in terms of its transfer functions [2]. It is supported by wreal in Verilog-AMS and real ports in SystemVerilog. By ignoring the inter-block feedbacks, there

is no need for matrix solution anymore and the RNM model can run in a pure digital event-based environment. As a result, the models may not be accurate enough for checking analog designs themselves, but it is fast and good enough for SOC simulation and any nightly regression test, ensuring that digital logics and analog blocks work together correctly without these "highly embarrassing" bugs. In addition to the speed benefits, RNM can also leverage digital verification methodologies, such as assertions, testbench automation, constrained-random stimulus, and Metric Driven Verification (MDV) [4].

RNM maybe a promising approach for mixed-signal SOC verification, but some questions remain unanswered:

- How to create the RNM model efficiently? What is appropriate level of abstraction?
- Do we really need to model everything from scratch, even though some of the blocks are mainly digital logics?
- How to make sure the RNM model is equivalent to the circuit and is always in sync with any design change?

In this paper, we propose the NVVM (Netlist-based Verilog Verification Methodology), a highly automated model generation flow, to solve these issues. The rest of the paper is organized as follows. Section 2 introduces the background of the Flash Memory IP where this flow is originated. Section 3 presents the details of this flow and discusses how analog signals are taken care of in a digital context. Section 4 demonstrates the results of NVVM.

II. BACKGROUND OF NVVM

A. *Flash Memory IP*

The NOR Flash memory array is composed of SST third generation SuperFlash® (ESF3) cells in cross-point structure, where the gates of the cells in one row are tied to the same wordline, and the drains of the cells in one column are tied to the same bitline. Figure 2 shows a simplified schematic of the ESF3 split-gate bit cell.

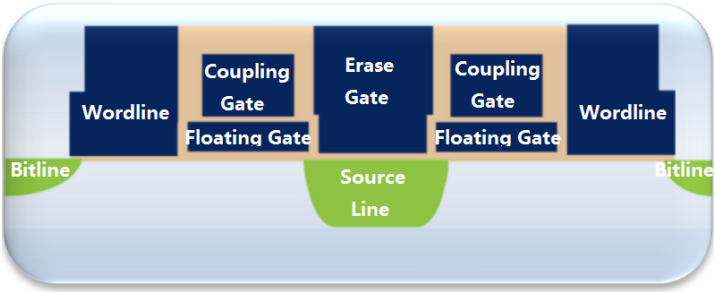


Fig. 1. Schematic of SST ESF3 flash memory cell [5]

During the Erase operation, logic 1's are put into a flash memory sector. A high voltage (>10V) is applied to Erase Gate (EG), while other terminals are grounded. The poly-to-poly Fowler-Nordheim tunneling between EG and Floating Gate (FG) pulls the electrons off the FG.

During the Program operation, logic 0's are put into flash memory bits. A high voltage (>10V) is applied to Coupling Gate (CG) to completely turn on the channel below FG, while the Word-line transistor (WL) is kept in saturation with a small voltage on it to pinch off the channel below WL. A large number of hot electrons are generated at the gap region between FG and WL channels, allowing the Source-Side Injection to FG.

The memory cell is read with a medium voltage on the WL and a reference voltage on the bit line (BL). The programmed cell has a higher threshold voltage and will thus conduct less than an erased cell at the same condition.

	WL (SG)	BL (Drain)	Source	EG	CG
Erase	0V	0V	0V	11V	0V
Program	1V	~2μA	4.5V	4.5V	10.5V
Read	Vcc	0.6V	0V	0V	Vcc

Fig. 2. Bitcell typical operating conditions [5]

The typical operating conditions of the bitcell are listed in Figure 2. From this table, it is obvious that managing a Flash memory array requires a number of voltage levels higher than Vcc, making it a complex mix-signal design. The main components of the Flash Memory IP are as follows:

- Charge Pump for the on-chip generation of high voltages.
- Analog level shifters, high-voltage drivers and power switches for power domain interactions.
- Current mirrors and reference voltage generator.
- Sense amp and timing control.
- State machine for mode control.
- Row mux and digital logics for address decoding.
- Redundancy mux to shift the main array to repair elements.
- Additional paths for test modes, measurement and signal override

B. Difficulty in Verification

The verification of this Flash memory IP has always been a big issue. The SPICE simulation of a 128KB Flash bank along with the charge pump takes weeks to months for some simple input vectors, and it may even crash in some simulators due to the large number of transistors. In fact, we need to employ some simplified netlist, such as breaking some feedback loop in the charge pump and replacing the bitcell array with equivalent resistors and capacitors. However, the simulation time is still in days, and the functional coverage is far from satisfactory.

To create a behavioral or real number model for this IP would be nontrivial. The design has a somewhat flat organization, and some large blocks have more than 100 transistors, 50 I/O ports and multiple functionalities. In some other blocks where consideration is given to layout quality and parasitic calculation, a single functionality may be split and realized in multiple blocks. In order to make sure the model is easy to understand and validate, additional levels of hierarchy need to be inserted manually, which involves great efforts, and the model synchronization and equivalence check would be a big challenge.

However, if we only want to verify the functionality of the Flash Memory IP, not all of the aforementioned components need to be modeled manually with accurate real numbers. The state machine, decoders, redundancy muxes and many configuration logics embedded in analog blocks are pure digital; the level shifters, power switches, test mode switches and some current mirrors are simple analog components that can be modeled with 0/1 signals; We may apply automated structural model generation to these simple and ubiquitous components, and we only need to write the behavioral or the real number model for a few complex analog components individually, such as sense amp, reference voltage generator and charge pump.

Figure 3 shows the tradeoffs of mixed-signal models. The area of bubbles represents the efforts needed to create the model. Compared to RNM, NVVM loses some accuracy by treating most signal as digital 0/1, but the automated generation requires much less modeling efforts and provides a much closer one to one mapping between the actual circuit and the netlist model. Compared to a pure digital black-box model, NVM retains the useful information of internal nodes while it's still fast enough for the SOC verification.

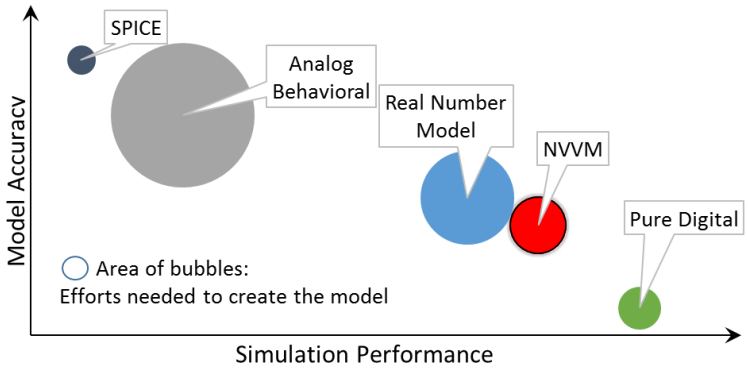


Fig. 3. Mixed-Signal Model Tradeoffs

III. DETAILS OF NVVM FLOW

The NVVM is shown in Figure 4. It does not require the great effort to create the model from scratch; most of the work is automated.

At first, a raw Verilog netlist is mapped from the transistor level schematic; the hierarchy and connections are kept exactly the same, except that most SPICE model parameters are discarded.

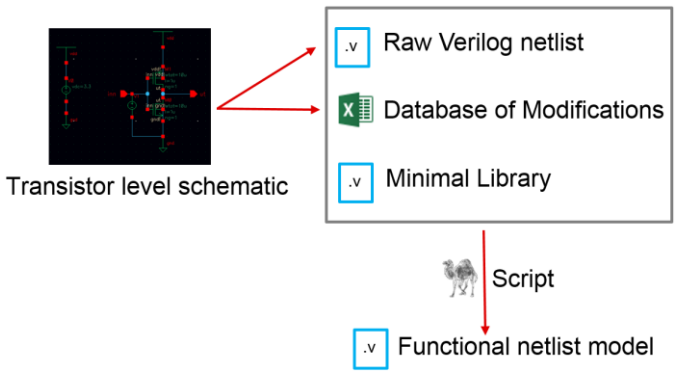


Fig. 4. Flow chart of NVVM

Secondly, a verification engineer or an analog design engineer inspects the schematic and maintain an Excel database, recording all the necessary modifications to make the netlist function in the digital environment. Similar to RNM, NVVM also employs the signal flow approach. In order to keep the signal from turning to 'X' and getting blocked somewhere, some modifications are necessary.

Thirdly, a verification engineer maintains the minimal library, defining different kinds of primitives we want to use in the modified netlist. They are merely simple wrappers of transistors and gates, with some modifier such as strength definition, delay and continuous assignments.

At last, a Perl script will execute the modifications and generate a functional netlist model with a simple testbench that is ready to run. The highly automated process ensures the synchronization between model and schematic.

The most essential part of work is to make sure analog netlist and signals function in a digital context. Some general rules of modification are listed as follows.

A. Identify signal direction

Verilog signal flow is unidirectional by default; the continuous assignment is unidirectional, and so are the pmos/nmos primitives, whose input is the S port and output is the D port. Therefore, it is important to identify the signal direction. For example, during programming the bitline of unselected bitcells should be raised to Vinh to decrease the program disturbance. In the analog mux in Figure 5, Vinh should thus act as an input source and be

connected to S, while LBL should be the destination tied to D. Analog designers typically do not care about this kind of issues because the drain and source are equivalent in SPICE simulation, but they need to be identified and fixed in a digital model.

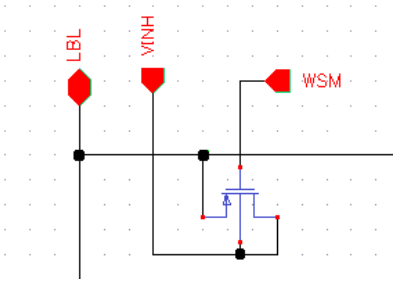


Fig. 5. A wrong signal direction in an analog mux

Bidirectional switch (tran*) should be used when signals may come in both directions. The WL switch of the bitcell is driver in Read operation and is driven by write driver in Program operation, so it should be a bidirectional switch. So are some duplex paths, which are used for measurement and external signal override in different test modes.

B. Resolve local contention/feedback

The original schematic has information such as transistor type, width, length and finger, and they will be used in the SPICE simulation to resolve the contention between different drivers on a wire. These information, however, is discarded by the netlister in the Verilog netlist, where every transistor is mapped to the same pmos/nmos primitive. Additional modification is needed for the correct behavior of the netlist model.

Figure 6 shows two examples. In (a), the long, weak PMOS loads (MP1 and MP2) need to yield to the strong NMOS, so that the output of the level shifter will follow the input. In (b), when the input is floating, the inverter pair holds the previous value; when the input is driven, the new value should win the contention over the weak feedback inverter and change the state. These weak transistors are recorded in the modification database, and they will be replaced by resistive switch (rpmos/rnoms) in the netlist.

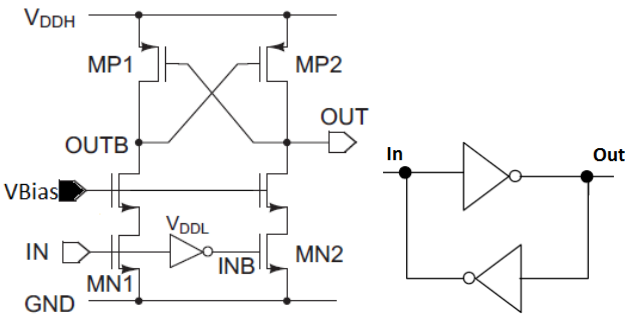


Fig. 6. Schematic of (a) a level shifter and (b) a inverter pair

There are also situations where a node need be forced to a certain value, such as reset, clear and precharge. The relevant transistors should have the highest priority and be marked as supply/strong in the modification database.

C. Initialize the feedback loop

In the real circuit, a well-designed feedback loop settles down very quickly upon startup. Inverter rings have a strong positive feedback and will start to oscillate. Pulse generator and frequency divider have a strong negative feedback so that they will stay unchanged at 0 or 1 if the enable signal is low. However, in the Verilog, there is no concept of loop gain, and all the nodes have an initial condition of 'z', leading to stuck-at-x in the whole feedback loop. Initialization or 'x'-filter is needed in such situations.

D. Indentify capacitive nodes

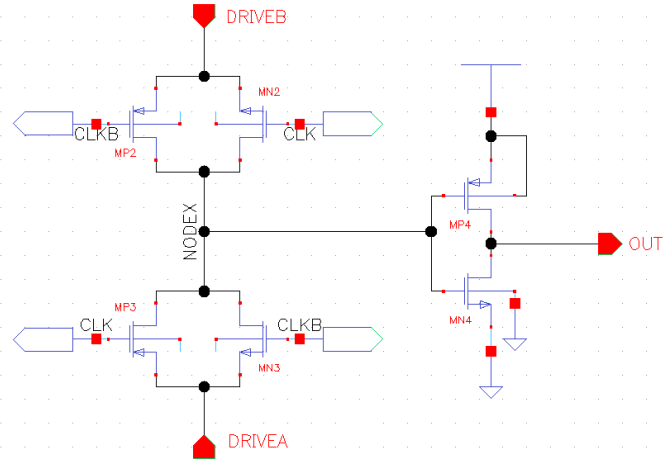


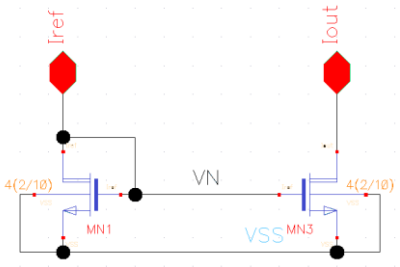
Fig. 7. Clock overlap produce an 'x' at the output

In the real analog circuit, every node has some capacitance and will hold the value for some time. In Verilog, however, a wire will become 'z' at once if there is no driver, and this could cause some issue. In figure 7, the non-ideal clock has skew so that clk and clkb may have a 1-1 overlap; during that period, both drivers are turned off and nodeX will be floating and populate an undesired 'x' at the output. To solve this issue, nodeX needs to be treated as a Verilog Trireg. In addition, trireg net is used to model charge storage nodes, such as dynamic logic and RC delay node.

E. Indentify diode connection

In NVVM, current source is modeled as 1/z for on/off, and current sink is modeled as 0/z. Diode connections are thus regards as a weak pullup/pulldown.

```
assign (weak0, weak1) D = S;
```



PMOS source	VN	NMOS sink
z (off)	0	z
1 (on)	1	0

Fig. 8. Modeling for a current mirror

For example, in Figure 8, when Iref input is off, VN will stay low and disable the Iout; when Iref is on, it will win the contention over diode and raise the voltage of VN by several hundred millivolts. VN will be logical 1 and then turn on the current sink, generating a 0 at the output.

IV. Simulation Result

We have tried NVVM on a 128KB Flash Memory IP and the result is satisfying. The testbench written for the spec-based model or the SPICE model can be employed by this netlist-based model. With NVVM, now we have got more information inside each block (Fig. 10) in addition to the top-level outputs (Fig. 9), while the performance is almost on par with a pure digital behavioral model and orders of magnitude faster than SPICE. Though the simulation log shows there are around one million instances, programming and reading the all the address of Flash Bank only takes several minutes. Furthermore, the netlist-based model also supports debug/test mode, allowing us to check whether a testpad input reaches the right destination and override the control signal as expected.

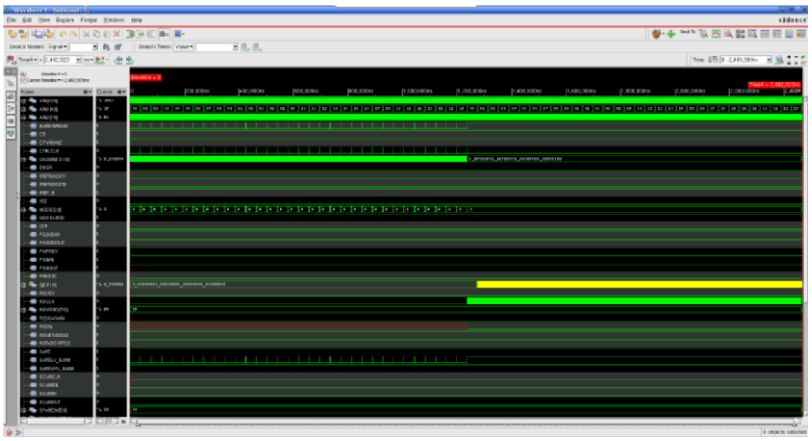


Fig. 9. The output of the testbench

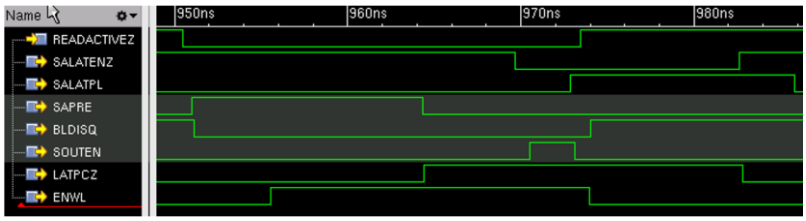


Fig. 10. A snapshot of some internal signals

Another advantage of NVVM is that the simulation is done in a pure digital environment, so many digital verification techniques can be employed. Figure 11 shows that the model is tested with random stimulus and assertions. We have managed to add back annotation and real number support in some critical blocks, and verification planning and coverage analysis for this model is also under development.

```
1. Powerup Initialization

2. Mass erase entire memory
  MASS ERASE: WEPROTECT 00000000 WEOTPPROTECT: 0 WEENGRPROTECT: 0
  MODE: f

3. Program random data pattern to all user memory addresses
  MODE: a
  PROGRAM: Address 0000 OTP: 0 ENGR: 0  DATA: 4c0895e818484d609b1f0566306b97b0d
  PASS: Signal                              USER Memory === 4c0895e818484d609b1f0566306b97b0d
  PROGRAM: Address 0001 OTP: 0 ENGR: 0  DATA: db2c284658937521200f3e30106d7cd0d
  PASS: Signal                              USER Memory === db2c284658937521200f3e30106d7cd0d
  PROGRAM: Address 0002 OTP: 0 ENGR: 0  DATA: 61e8dcd3d76d457ed462df78c7cfde9f9
  PASS: Signal                              USER Memory === 61e8dcd3d76d457ed462df78c7cfde9f9
  PROGRAM: Address 0003 OTP: 0 ENGR: 0  DATA: 6e2f784c5d513d2aa72aff7e5bbd27277
  PASS: Signal                              USER Memory === 6e2f784c5d513d2aa72aff7e5bbd27277
  PROGRAM: Address 0004 OTP: 0 ENGR: 0  DATA: 247ecdb8f793069f2e77696cef4007ae8
```

Fig. 11. Assertion based verification with NVVM model

To conclude, we think NVVM is a sweet spot for fast function verification of mixed-signal circuits. It's fast, and it provides enough details. The highly automated flow makes the generation of model transparent to the analog designers, so they can be also actively involved in the verification. The chance of misunderstanding between the real circuit and abstract model is minimized.

[1] Karnane, Kishore, G. Curtis, and R. Goering. "Solutions For Mixed-Signal Soc Verification." Cadence Design Systems (2009).

[2] Hartong, Walter, and Scott Cranston. "Real valued modeling for mixed signal simulation." Cadence Design Systems (2009).

[3] August, Nathaniel. "A Robust and Efficient Pre-Silicon Validation Environment for Mixed-Signal Circuits on Intel's Test Chips." Quality Electronic Design, 2008. ISQED 2008. 9th International Symposium on. IEEE, 2008.

[4] Khan, Neyaz, Yaron Kashai, and Hao Fang. "Metric Driven Verification of Mixed-Signal Designs." Design and Verification Conference (DVCon). 2011.

[5] Alexander Kotov, "Three generations of Embedded SuperFlash split gate cell: scaling progress and challenges for future technology nodes", Memory Workshop 2013

[6] Andy Milne, Damian Roberts. "Utilizing Digital Techniques for Analog and Mixed-Signal Verification." Synopsys, Inc (2011).

[7] Kundert, Ken, and Henry Chang. "Verifying all of an SOC-analog circuitry included." Solid-State Circuits Magazine, IEEE 1.4 (2009): 26-32.