Application hardware simulation to optimize system functions

Chapter 1: Introduction

Full-chip simulations are very quickly limited by excessive simulation time as well as by the availability of appropriate models. In the same way, simulation of a system cannot be effectively performed at the most accurate level both because it requires too much simulation time and because multi-domain systems cannot be accurately modeled using a single design language. Furthermore, when integrating components into a system, the system designer must necessarily consider that they meet the specifications and that they individually work as specified. The focus is then on verifying that the assembly in the system works as specified.

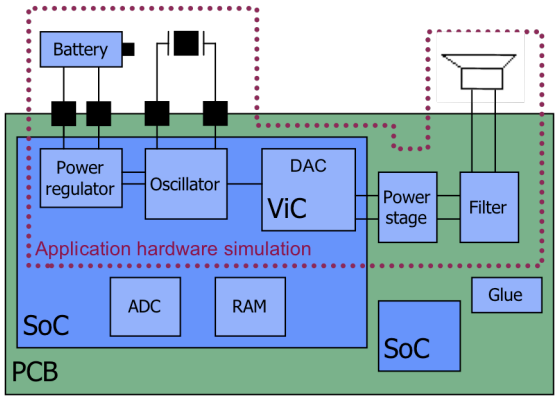
The stake is therefore to use a methodology that will bring the best time/accuracy ratio. As indicated, performing a full-chip simulation is unrealistically long and deducing the performances of a complete system from the performances of each individual component is not accurate as it does not take into account the sensitivity to distortions and disturbances.

This article aims at explaining a new approach to simulation that consists in simulating functions of a system to check and optimize performances; it is called "Application Hardware Modeling" (AHM). We will first give the objectives, the steps and the benefits of the AHM approach. Then, we will highlight its use through Pop-up Noise simulation and, to finish, we will open towards other possible applications.

Chapter 2: What is AHM?

2.1 What are the objectives of the AHM approach?

The goal of the AHM approach is to optimize an overall function performed jointly by parts of the system, comprising some Virtual Component (so called ViC or silicon IP blocks) within a SoC assembled on the PCB with discrete components, such as Quartz, PMIC, or MEMS, along with application software.



Therefore, contrarily to so-called full-circuit or full-chip simulation approaches, Dolphin Integration through its multi-level simulator SMASH promotes the Application Hardware Modeling (AHM) approach:

- While traditional simulation focuses on validation of each component of the system, AHM simulation focused on validation of the assembly of the components to build-up the system function and taking into account introduces distortions and disturbances.
- While full-chip simulation is performed with fast analog simulators using the most accurate possible description in SPICE with extracted parasitic devices, AHM simulation is performed using the most appropriate description level for each part of system, whether structural analog in SPICE, behavioral analog in HDL-AMS, gate-level, RTL or behavioral logic in HDL…
- While full-chip simulation is performed on the chip as a subset of the system, AHM simulation is performed on a specific function as a subset of the system.
- While full-chip simulation is restricted to the parts of the system that can be modeled at the structural level in SPICE, AHM simulation can include multi-domain parts such as MEMS, microphones, actuators… that are needed to fully simulate and verify the behavior of the function.

AHM addresses the need to simulate functions of a system, which is currently not addressed by traditional design flows and tools.

2.2 What are the steps to perform an Application Hardware simulation?

To apply the AHM approach, the designer has to define the application function that is to be checked in the system, whether audio, video, etc. All the components involved in the function should be identified. The corresponding application schematic must then be designed and assembled using the models of each element at the appropriate description level required to perform the verification of the function performance:

- for the required parts of the ViC, supplied by the ViC provider;
- for the required parts of the SoC, supplied by the SoC Integrator;
- for the additional ICs and discrete components implanted on the PCB, supplied by the IC vendors and standard part providers, usually in SPICE;
- for the MEMS, actuators, loud speakers, headphones…, modeled directly by the application engineer, optionally using dedicated EMBLEM libraries;
- …

In most cases, the Field Application Engineers (FAE) of the designer's company and of its suppliers will be involved in the AHM modeling in order to appropriately choose the model description levels and assemble the most relevant application schematic.

As the system function goes through the ViC, the SoC and the PCB, the application schematic might include both electronic and electro-mechanical parts. Therefore, to perform an application hardware simulation, the simulator needs to support HDL/HDL-AMS models and must be truly mixed-signal. Moreover, the simulator must have multi-level capabilities in order to model each part of the function at the appropriate level for best accuracy/speed tradeoffs.

2.3 What are the benefits of this approach?

Performing an application hardware simulation allows the designers and/or FAEs to anticipate the disruption linked to the integration of components in the system. When performing the selection of components to integrate into the system, designers naturally consider the performances given in the specification but do not sufficiently take into account the disturbances resulting from their assembly in the application.

The AHM approach implies the simulation of the application schematic of a complete function. As this simulation is performed from the models of the components, it is easy and convenient to replace, in the schematic, the model of one component by a similar one to check the impact on the performances of the function and thus optimize the application schematic.

Identifying a design defect in a SoC at the system level is very expensive. It may, at the very least, increase the Bill-of-Material (BoM) cost but can also involve a SoC re-spin or redesign.
Using an AHM approach, problems can be identified earlier in the design flow thanks to the simulation of the function. Furthermore, the design can be optimized by avoiding excessive design margins. This helps reduce costs as well as time-to-market market of the final application.

We will now go through the example of Pop-up Noise simulation to highlight the use and the advantages of the AHM approach with a real application.

Chapter 3: AHM approach for Pop-up Noise (PuN) analysis of audio applications

3.1 Introduction to PuN and its quantification

Pop-up noise (PuN) is referred to for any undesirable transition noise or audible glitch occurring when the audio path switches from one operating mode to another. This phenomenon damages the end-user listening comfort and reduces the perceived quality of the audio system, even though it cannot be accounted for in the current audio industry definition of SNR or THD (the most common audio performance indicators).

While the PuN issue becomes a strong audio product key differentiator, only a few expert companies have started to address the objective PuN quantification and distinguish themselves from other unaware suppliers who content themselves by claiming that their architecture is "pop noise free" or "clickless/popless operation".

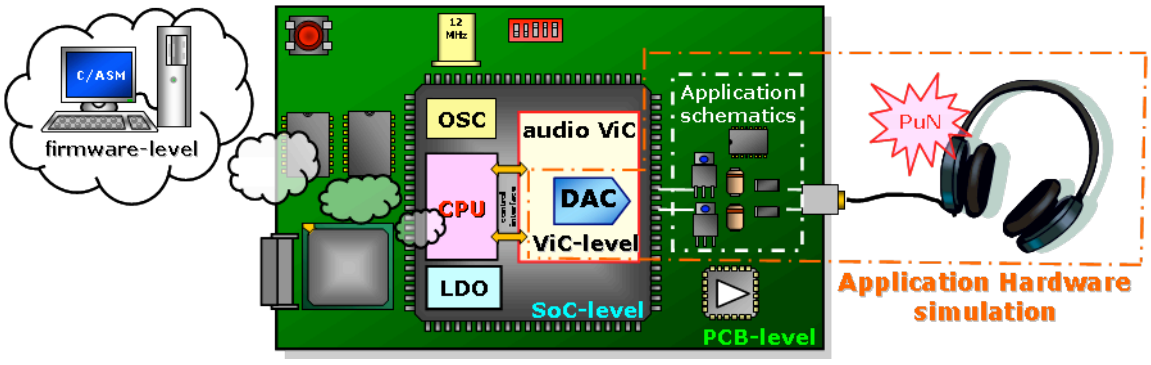
Dolphin has been the first ViC supplier to succeed in simulating and quantifying PuN, together with the application schematic of the innovative audio CODECs. In our ViC specifications, the PuN is specified (for active <-> inactive mode) together with its description and benchmark. As our simulation results are calibrated against silicon measurements, through our Virtual Fab Process (VFP), we may commit to having the PuN figure meet our specification. Recommended sequences (start-up/power-up, changing mode) for PuN reduction are also part of our specifications.

PuN is defined by the ratio of its peak amplitude after A-weighted filtering on the Full Scale output signal peak amplitude. It is considered as audible if it is above -50 dB and acceptable for the value below -60 dB.

PuN may have multiple sources at ViC, SoC and PCB levels. Dolphin has strived to identify the possible causes of PuN and keep innovating for reducing it. The results of these efforts are delivered in two patent-pending solutions for PuN reduction since the beginning of 2009.

Simulation for PuN reduction and identification of PuN causes can only be achieved using the AHM approach, with a truly multi-level and multi-domain simulator such as SMASH™!

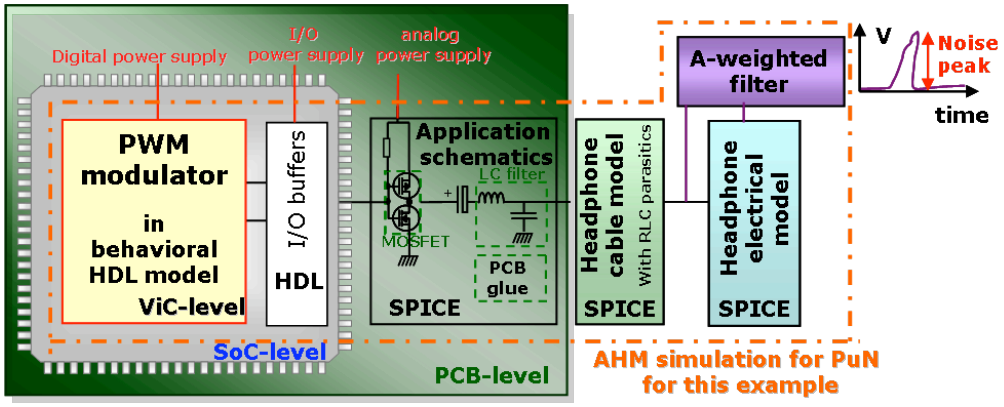
By performing an AHM simulation with SMASH™ for PuN analysis, you will be able to quantify the PuN of a given audio system including ViC, SoC and PCB level, per a given application schematics configuration (ASC). You can then smoothly optimize or switch between different ASC and/or different application programs/firmwares for PuN reduction. All that without fabricating several tests boards or different daughter boards for each ASC, while saving time and resources from costly diagnostics!



3.2 Presentation of the simulation environment example for AHM for an audio component

We explain in detail the advantage of AHM simulation for PuN through an example using an audio Class-D DAC with pure logic PWM modulator in SoC and external on-board MOSFET power stage and LC filtering.

The AHM simulation environment is shown in figure below. It comprises the model of the digital audio ViC, I/O buffer, the applications schematics on the PCB, model of the glue for PCB routing parasitics, model of the headphone cable (resistive and capacitive parasitics) and finally the model of the headphone.
Note that depending on your simulation solution, digital to analog interface devices may need to be inserted into the netlist in order to perform a true mixed-signal simulation. Naturally, when using SMASH™, the required interface devices will be automatically identified and inserted without any intervention from the designer other than correctly configuring the interface device characteristics.



The PuN is simulated after A-weighted filtering of the peak transient voltage received across the headphone load. For the example, the PuN is quantified during two operating mode transitions of the audio ViC, supposing that the power supplies of the system are stable and that power supply noise does not inject further unwanted audible noises at the headphone output. This is in the aim of simplifying the example.

Please note that, if the audio ViC is a mixed-signal component, in order to take into account the SoC level influences on PuN, the process variation per a technological process and the I/O buffer should be modeled appropriately as the digital to analog conversion is performed by the ViC, thus in the SoC.

For the example of a digital audio solution, the digital to analog conversion is done on the PCB. Therefore the model of digital audio ViC and the I/O buffer can be modeled in behavioral logic HDL language. The other models are defined in analog SPICE.

Please note that, if we want to define the PuN at system power-up, then the power supply regulator of each part shall be well modeled to correctly predict the PuN.

For this example, we can use stable and ideal power supplies.

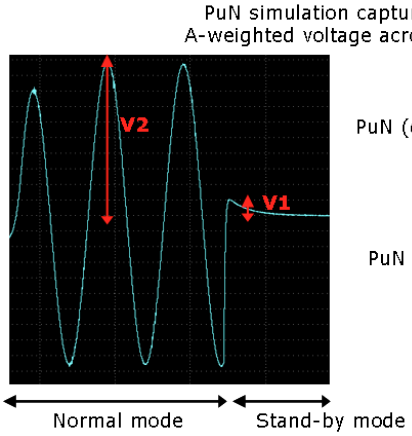
3.3 Description of the PuN test sequences

The test sequences are for two operation mode transitions of the audio ViC: one from normal mode to soft-mute mode, the other from normal mode to stand-by mode.
The normal mode is the normal operation mode for audio playback.
The soft-mute mode is the mode which progressively reduces the volume of an audio file to minimum; the PWM modulator continues to modulate as the input signal is zero. The steps of the volume reduction could be programmable.
The stand-by mode is the mode where the input of the PWM signal is put to zero as the mode is activated; the PWM modulator continues to modulate as the input signal is zero.

The input test signal is a Full Scale 1 kHz sinewave. The PWM bitstreams are generated by the audio ViC and output on PCB for connecting to the ASC.

The PuN simulation results show that:

- for the transition from normal mode to soft-mute mode, there is no PuN at all (for the possible programmable volume reduction steps),
- for the transition from normal mode to stand-by mode, the PuN is loudly audible and it is around -20 dB, far above -50 dB.



$$\text{PuN (dB)} = 20 \log \frac{\text{Transient noise peak amplitude (A-weighted)}}{\text{Full Scale output channel peak amplitude}}$$

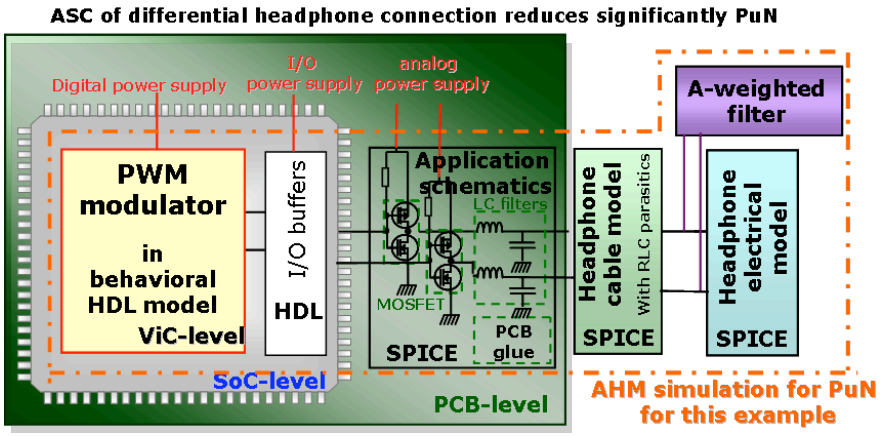
$$\text{PuN (dB)} = 20 \log \frac{V1}{V2} = -20 \text{ dB} >> -50 \text{ dB}$$

3.4. PuN reduction with AHM simulation

Once the transition which introduces a PuN above the PuN threshold is identified (here from normal mode to stand-by mode), different approaches for PuN reduction can be envisioned.

For example, by modifying the ASC of the audio ViC, we could use a differential headphone connection instead of single-ended headphone connection.
Note that optimizing the ASC for PuN reduction could impact on other system performances or cost. Sometimes, an optimized ASC for PuN reduction can have a higher BoM cost than the reference one and/or can be an ASC which is not optimized for overall system SNR performance. Finding the most appropriate ASC for a given audio ViC is always the result of a trade-off between different priorities for diverse criteria (e.g. PuN, SNR, power consumption…).

We can see that for differential headphone connection, there is (almost) no PuN observed, as the same glitch is passed through both differential signals, (+) and (-), so the difference is still null.
Note that it depends on the tolerance of the capacitors used for the LC filters and the PCB parasitic characteristics. If the PCB routing of the differential signals is too different, or there is some delay between the two differential signals, then the glitch amplitude could be slightly different on the differential signal: in this case a slight PuN can still be observed, or 2 glitches are observed instead of one due to the delay. These parameters depend mainly on the PCB design.



Another approach for PuN reduction could be focused on the software/firmware application.
Here, instead of going directly from normal mode to stand-by mode, a combined transition from normal mode to soft-mute mode followed by soft-mute mode to stand-by mode can be introduced in the application software.

It can be predictable that this software optimization does remove the occurrence of PuN. Indeed, no PuN occurs for the transition from normal mode to soft-mute mode. And, as long as the duration of the transition from soft-mute mode to stand-by mode is carefully configured, no PuN occurs for this transition. This duration would depend on the input signal amplitude and the volume reduction step. It can be easily verified by simulation.

For this example, by considering the pros and cons of the two PuN reduction approaches, the software optimization can be the best way as it does not use further CPU resources and requires no changes to the ASC (or for applications where a differential headphone connection is not possible).

Please note that the simulation results are compliant with the measurements done with the equivalent PCB platform.

3.5 Evolution of AHM for PuN

The modeling of the example or the PuN reduction proposal described in the example might seem obvious or over simplified, yet the principal idea is applicable for many more complicated situations. The AHM approach for PuN analysis enables PuN optimization either at ViC, SoC, PCB or software/firmware level. PCB routing characteristics and different ASC can be determined for minimizing PuN prior to the PCB design so there is no need to re-fabricate several PCB or play the "try and/or die" on the PCB by changing components, connecting flying-wires…

The near future evolution of AHM for PuN could be the mechanical modeling of headphone jack connectors and of the headphone transducer (instead of an electrical load). As the insertion of headphone jacks can cause PuN, and depending on the sensitivity of each headphone, the PuN sound perceived could be different. The definition and quantification of the PuN can also evolve thanks to the AHM approach.

AHM for PuN analysis can be part of Dolphin's application engineering service provided together with innovative audio CODECs and all-in-one SMASH™ simulator. Application engineers of our customers can thus have this secret weapon to save time and money!

Chapter 4: Other functions and application opening for AHM approach

In the previous part, we illustrated the benefits of PuN simulation to optimize the application schematic of a system and reduce the cost and time-to-market of this system. However, it is important not to forget that the benefits of the AHM simulation with SMASH™ are not limited to PuN or even to audio functions.

Any number of function behavior and performance validations can be performed using an AHM approach, among which:

- analysis of the influence of Jitter on the application function,
- analysis of power consumption peaks for noise reduction in the supply regulation,
- analysis of power consumption in the different modes of the application to reduce overall power consumption of the system,
- analysis by simulation instead of using bread boards to reduce evaluation costs while enabling exploration of different application schematics,
- analysis to verify data rates and check that the design meets speed requirements,
- analysis to help select an optimized solution for power management to avoid wasted area or higher costs,
- analysis to detect sensitivity to disturbances and to quantify performances (SNR…),
- …

Depending on the targeted performance validation, a specific application hardware model may be needed with different requirements for the component description levels. The appropriate description levels of the components depend on the required accuracy as well as on the behavior and effects that must be modeled in order to perform the validation of the function.

AHM simulation can also help to define an application chipset with the selection of appropriate components. Indeed, an application schematic can be optimized by switching between the HDL models of some components. After simulation, the best configuration of the application schematic can be identified: for best performances, for lowest BoM cost…

Chapter 5: Conclusion

This paper highlights a new simulation approach which addresses a need that is currently not addressed. It also presents the unique capabilities of SMASH™ for simulation of the application hardware model (AHM) of a system specific function by combining the models of each component of the function at the appropriate description levels required for the analyses. The goal of such simulations is to simulate the overall function to detect design defects in the assembly and to verify the function performances.

SMASH™ is the best solution for AHM simulation because it natively handles multiple description levels with several hardware description languages in a single netlist. Moreover, SMASH is known as a reference for its VHDL-AMS compliance and support while EMBLEM libraries supplied by DOLPHIN are the basis for AHM model assembly.