SUCCESS™ for embedded soft and hard coverification of SoC

by Ann MAGIORANI and Pierre SAUGE
Tel-Aviv. April 2003

Abstract

Covalidating hardware and software remains a complex task for SoC designers. Isolated Instruction Set Simulators (ISS) and hardware simulators are readily available from diverse suppliers, but none can offer a complete solution for simulating a whole SoC with its software application: i.e. simulation of the microprocessor together with interconnected peripherals: logic, analog, memories… which may be described in VERILOG-HDL, VHDL, SPICE, VHDL-AMS, C-language…

This article presents an innovative solution to fill the EDA gap between hardware and software designers allowing a sophisticated coverification between the application program of a microcontroller with its analog or logic peripherals.
It is based on a cosimulation methodology, SUCCESS™, using the debug capabilities of an Integrated Development Environment (IDE) combined with those of the mixed signal hardware simulator SMASH™.
The major benefit is not just to allow Virtual In-Circuit Emulation (ICE), i.e. the ICE debug capabilities with an electronic simulator at software level, but to provide the ICE capability at the beginning of the design flow, which drastically improve the Time-To-Market.
SUCCESS™ thus becomes the favorite solution to speed-up check-out at three stages of the SoC design flow: the choice of architecture, the SoC hardware configuration and the application program to embed in a SoC.

After an explanation of the principal difficulties faced by the hardware and software designers, this paper presents an efficient method for coverifying a mixed signal System-on-Chip with its application program.
As a conclusion, it presents the implementation of this method on a mixed signal example.

Paper

ICE capabilities ahead of hardware!

While the advantages of a Virtual Test Methodology (VTM), i.e. codesign of the test program and the circuit on an electronic simulator, is used for reducing Time-To-Market of SoC designs, innovative solutions are emerging to increase designers' productivity thanks to complete hardware / software codesign.

In a traditional design methodology, software and hardware are designed in parallel, with connection for the prototype: no convenient tools were existing to fill this missing EDA link. We saw recently the emergence of some tools for logic applications, but the mixed-signal domain (70% of the System-on-Chips are planned to be mixed-signal in 2007) is still difficult to handle.
On the one hand, hardware designers relish validating, at system level, whether the microprocessor is correctly connected to each peripheral, but also validating the functionality of peripherals in their SoC environment.
On the other hand, software engineers use Instruction Set Simulators (ISS) to debug their embedded application. These simulators are included within a fully Integrated Development Environment (IDE) that provides integration and access to all the professional developer's tools, from editing to compiling, linking, and debugging of embedded software.

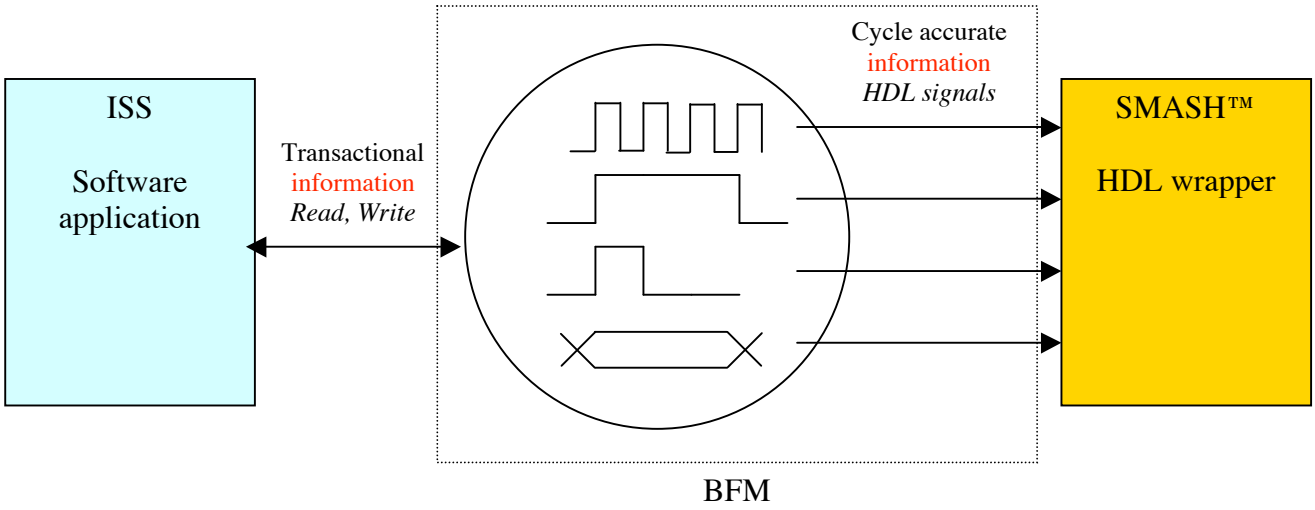
Isolated ISS and hardware simulators are readily available from diverse suppliers. They present differences in simulation speed, functionalities and ease of use. But none can offer a complete solution for simulating your whole SoC with its software application, i.e. simulation of the microprocessor together with interconnected peripherals: logic (I2C, SPI…), analog (ADC, DAC, amplifiers, PLL…), memories… which may be described in VERILOG-HDL, VHDL, SPICE, VHDL-AMS, C-language…

This leads to many manual adaptations: when an error is detected, engineers need to return to both the hardware and the software design teams. Thus, several prototypes are needed, which is cost ineffective and results in a long Time-To-Market.

The dream of a hardware / software codesign would be to enable the capability to develop application code taking into account the hardware development, as well as the reverse, which is a tough challenge since they are from different worlds and are not used to talking to each other. The benefit of this codesign capability would be to eliminate runs and reruns of prototypes to set-up the application to the hardware design.

1. The Dolphin SUCCESS™ environment

Codesign using the Dolphin SUCCESS™ environment is the solution to link the SMASH™ mixed-signal, multi level, mixed language simulator with any Instruction Set Simulator for an efficient co-verification allowing the debug of software running on simulated hardware.
SUCCESS™ is a consistent extension which is plugged between an IDE and SMASH™ in order to connect an instruction-accurate processor specific simulator (ISS) to a cycle-accurate simulator (SMASH™) through a Bus Functional Model[1] (BFM) of the processor.

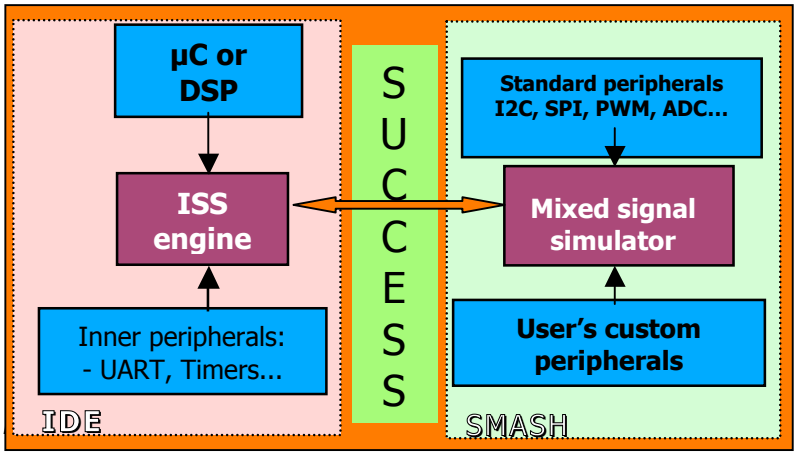


2. What are the advantages?

HW/SW coverification brings great benefits to reduce risk of critical design errors found late in the design process.
It combines the advantages of the IDE with debug capabilities with those of SMASH™ mixed-signal simulator, i.e. the co-validation between the application program of any microcontroller with any peripheral, would they be logic, analog or mixed signal.

[1] The Bus Functional Model (BFM) is a module that simulates the behavior of a microprocessor as observed externally on the bus. It allows SUCCESS™ to transform transactional information which is provided by an ISS, as a read/write access to a memory in a cycle accurate bus access through an HDL wrapper in our mixed simulator. It thus simulates faster than a full functional model, while still having the capability of emulating whatever behavior is necessary to exercise the logic under test external to the processor.

Since the peripherals are simulated in SMASH™, mixed signal, multi level simulator, there is no limitation in the capabilities of SUCCESS, which is the sole solution for simulating peripherals modeled in VERILOG-HDL, SPICE, VHDL, ABCD (C), VHDL-AMS!
The Virtual Test Bench of the microcontroller can be used at the soft, firm and hard level directly using the multi level capabilities of SMASH™, i.e. behavioral in HDL language, functional in RTL description netlist, and electrical for transistor level.



SUCCESS™ allows designers to save costly time of their design flow while increasing the chance of a success at the first pass!

SUCCESS™ can be used at three stages of your design flow.

1. SUCCESS™ when validating the choice of architecture for:
- verifying of the hardware/software partitioning
- selecting the micro-controller options the most adapted to the application
- choosing the mixed signal peripherals
- checking the interactions between the microcontroller and the peripherals

For this application, high level models are generally used.

2. SUCCESS™ when validating the SoC hardware configuration

It allows to:
- check-out the interconnections of mixed signal peripherals with the microcontroller
- verify that each ViC or block still functions within the SoC
- check-out of the different working modes of the SoC (including its test modes)
- development and check-out of test routines

A mix of high level and pin-accurate simulation models are used, short test cases must be developed in assembler or C.

3. SUCCESS™ when validating the application program to embed in a SoC
This is particularly important for:
- checking-up of the application software within the SoC
- exhaustive exercising for asynchronous events
The Application Program is run with Virtual Tracing as a preview of its usage with an Emulator.

Based on the first uses of SUCCESS™, this methodology improves design team productivity by reducing the validation stages.

3. A solution for all Integrated Development Environments (IDE)

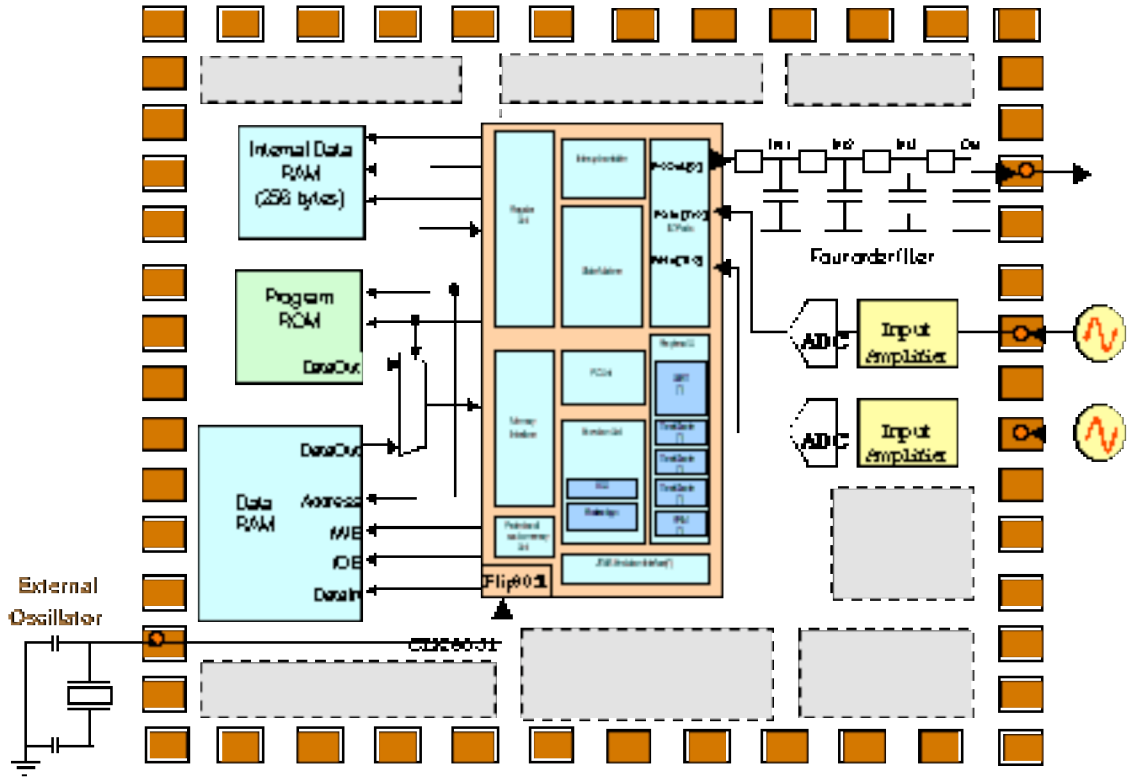
SUCCESS™ has been first developed for Dolphin's Flip8051 family of Virtual Components, with the IDEs available from Keil and Raisonance.
This methodology can be enhanced easily for the 8051 µP family but also for any microprocessor by the adaptation of the SUCCESS™ interface: nothing has to be changed in the ISS nor in SMASH™.
The adaptation can be done with any IDE which provides a Peripheral Development Kit (PDK) and an Application Programming Interface (API). SUCCESS™ can also be adapted to any of the ISS processor target families integrated in the IDE. Moreover, for a given processor target family, SUCCESS™ can use specialized BFM according to the exact bus specification of the processor.

4. SUCCESS™ through a significant example

The case study below highlights the benefits of SUCCESS™ when validating the SoC hardware configuration as well as the application program to embed in the SoC for a right-on-first-pass application.
The impact of the modification on the application program in the behavior of the logic and analog peripherals is illustrated, as well as the tuning of the fourth order filter for the hardware side.
For this case study, the SoC architecture has already been chosen and validated; the check-out of this architecture using SUCCESS™ is outside of the scope of this example.

The SoC below is composed of an 8051 microcontroller with its associated memories (ROM and RAM) and two ADCs.
The function of this SoC is to make some calculation on the two wave signals, with different frequencies and amplitudes, received and converted by the two ADCs.

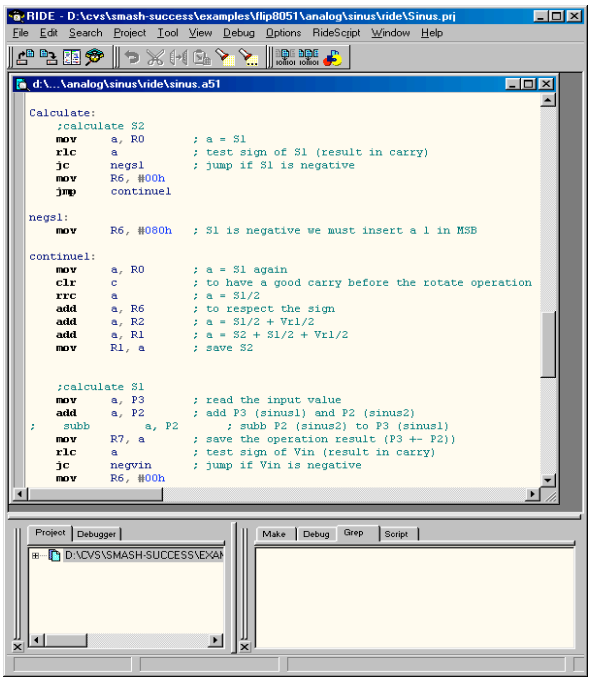
The addition and the subtraction of the two Sinus, sampled and converted into logic data, are computed by the 8051 microcontroller which generates the value of the bit stream on its port P0OUT[0] for each sample of the Sine signals.
The value is then filtered successively four times with four specifics RC filters, in order to eliminate the current noise and obtain a perfect sine, resulting from the computing done by the microcontroller.



Validating the software to embed in the SoC

At this stage, SUCCESS™ enables:
- to benefit from the Integrated Development Environment
- to debug the application program thanks to the debug capabilities of the IDE
- to modify in real time the calculation (addition vs. Subtraction) performed by the microcontroller, and view directly the results thanks to the simulation.

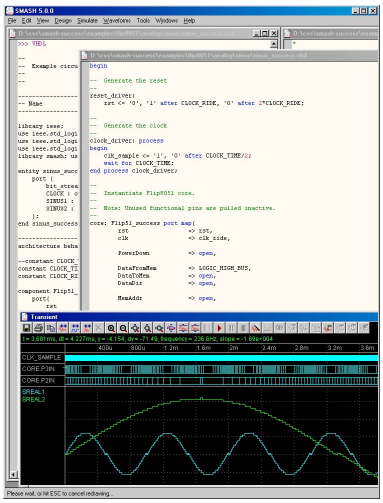
The application program is described in assembler code. It executes a combination, addition or subtraction, of two sinus waves with different frequencies and amplitudes.
It then computes and generates the value of the bit stream on pin "P0Out[0]" for each sample of the sinus signals.
For this kind of application, SUCCESS combines the flexibility of an IDE providing a user-friendly C or assembler source code development and debug environment with the display capability of a multi-level simulator.



The instruction accurate model simulated by the ISS allows the programmer to develop the application code for any analog or mixed signal peripheral while still benefiting from a well-known development environment.

Validating the SoC hardware configuration

Now, SUCCESS™ is used for:
- the validation of the configuration of the SoC
- the tuning in real time of the fourth order filter
- the check-out of the interconnectio between the microcontroller and logic and mixed-signal peripherals.



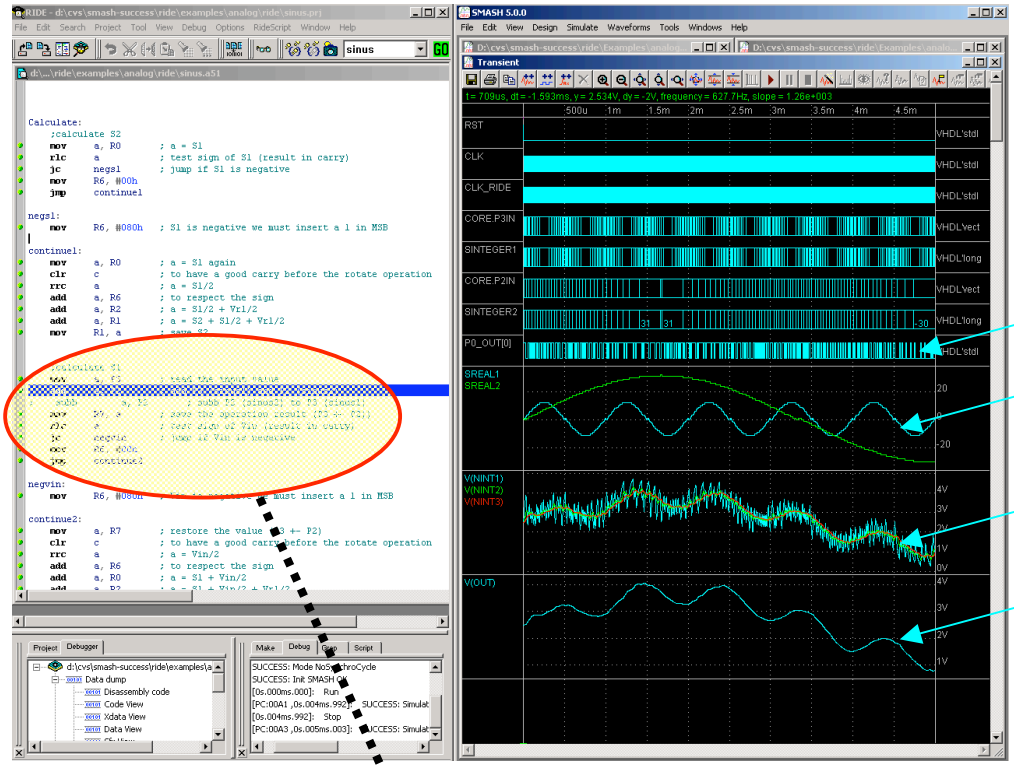
In this example, the peripherals are: two ADCs and one fourth order filter. The two ADC models digitize two sinus input signals and generate an hexadecimal value for each sample of the two sinus signals.
The bit stream generated by the application program on port P0Out[0] is filtered successively with four specific RC filters, in order to reject noise.
The interconnections between these peripherals and the microcontroller are then easily checked out with SMASH™ simulator.

Impact of the modification of the application program on the behavior of the peripherals

First case:
In the application program, the designer chooses to simulate the addition of the two sinus signals.

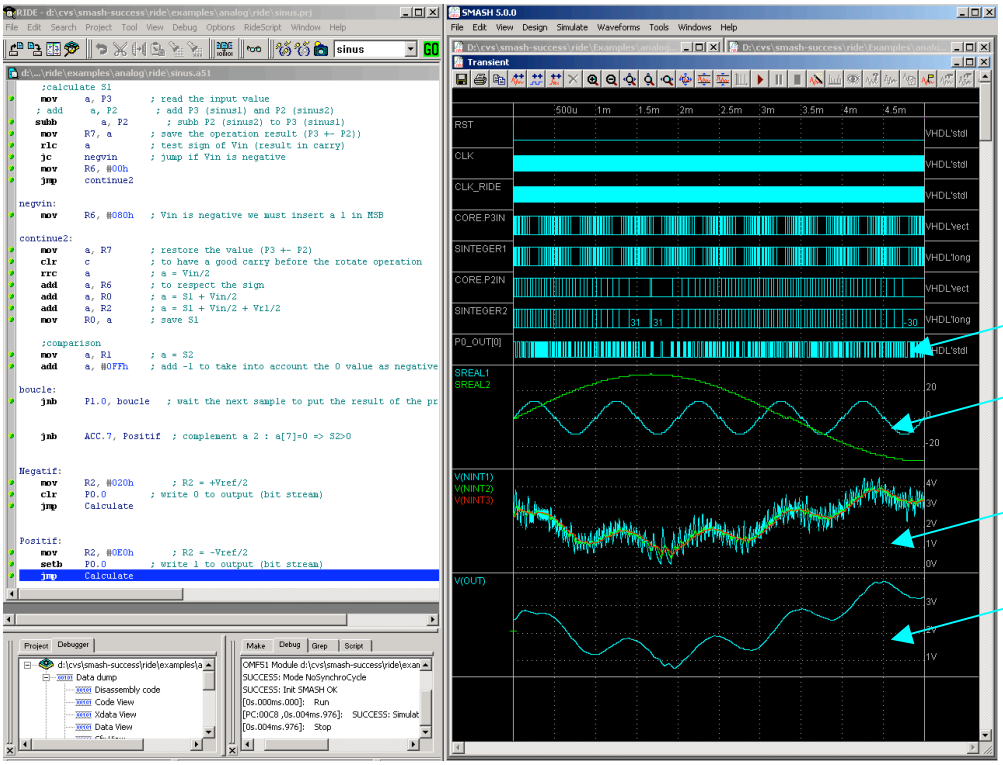
The validity of the bit stream can be easily checked thanks to the display of the filter output while saving time when compared to a simulation of the RTL model of the microprocessor.

SUCCESS™ allows the SoC designers to coverify the application program with the microcontroller and the peripherals.
Software and hardware correction of bugs can be done virtually to set-up the filters for the peripherals or making some sampling of the input signals.




```
    ;calculate S1
    mov     a, P3       ; read the input value
    add     a, P2       ; add P3 (sinus1) and P2 (sinus2)
;    subb       a, P2       ; subb P2 (sinus2) to P3 (sinus1)
    mov     R7, a       ; save the operation result (P3 +- P2))
    rlc     a           ; test sign of Vin (result in carry)
    jc      negvin      ; jump if Vin is negative
    mov     R6, #00h
    jmp     continue2
```

Second case:
Changes are made in the application program, where the addition is replaced by a subtraction. The modification is automatically taken into account and the designer needs only to re-run the cosimulation to validate the impact of those changes.



This case study shows the advantages of using SUCCESS™ for the validation of a simple SoC.
Advantages will be even more important for complex SoC architectures where interactions between embedded software and peripherals are much more difficult to validate, and where development of application code connected to hardware will harness the full power of SUCCESS.

The more complex the SoC is, the more SUCCESS becomes valuable.

SUCCESS emerges progressively

The acquaintance with such an integral solution as SUCCESS is facilitated by the difference with its would-be competitors. The ISS does not need to be reformatted prior to its linking with SMASH, so that the programming expert, and the electronic expert and the test programmer, either for analog or for logic peripherals, may converge on SUCCESS, each with his customary style when using the solutions separately.

References

[1] http://www.dolphin.fr/medal/success/success_overview.html
[2] http://www.dolphin.fr/medal/smash/smash_overview.html
[3] http://www.dolphin.fr/medal/smash/platinum/smash_week6.html
[4] http://www.dolphin.fr/flip/logic/logic_overview.html
[5] http://www.raisonance.com/
[6] http://www.keil.com/