# A Novel Approach to Reduce Leakage Power in GALS System architectures

A. Rajakumari
J.N.T.U.H
Hyderabad
Andhra Pradesh, India

Dr. N. S. Murthy Sharma
J.N.T.U. H
Hyderabad
Andhra Pradesh, India

Dr. K. Lal Kishore
J.N.T.U. H
Hyderabad
Andhra Pradesh, India

## ABSTRACT
Globally asynchronous locally synchronous (GALS) system architectures are known for low power consumption through clock gating techniques. In GALS architectures set of logical synchronous modules will communicate with other through asynchronous wrappers. Though this technique results in good dynamic power consumption, as the process technology shrinking down to 45nm and below the leakage power is equivalent to dynamic power consumption. In this paper, we are proposing a power gating technique for GALS architectures which uses existing handshaking signals of asynchronous wrappers to reduce both dynamic and leakage power consumption. To prove the proposed architecture we have implemented a GALS asynchronous micro controller from Daltons[1] synchronous 8051. For this we used Synopsys SAED 90nm library for synthesis and demonstrated the new proposed power gating control techniques through U.P.F (Unified Power Format) based simulation results.

## General Terms
Asynchronous Design, Leakage Power Reduction, and Dynamic Power Reduction.

## Keywords
GALS (Globally asynchronous locally synchronous), Power Gating, Power Gating Control, U.P.F (Unified Power Format), Clock Gating, 4-Phase Hand Shaking.

## 1. INTRODUCTION
The absence of global clock allows VLSI asynchronous circuits to offer several advantages over their synchronous counterparts such as low power and high speed. There are many reasons to implement a design using asynchronous technique. Asynchronous designs have the advantages over traditional synchronous designs of lower power consumption, no clock skew, better technology migration, and less global timing issues [2]. There are numerous ways to implement an asynchronous circuit; these include fundamental mode Huffman circuits, burst-mode circuits, and Muller circuits. Digital circuits in today's commercial products are almost exclusively synchronous. However most of the digital circuits in today's commercial products are almost exclusively synchronous. There are several EDA (Electronic Design Automation) tools available in the market to implement synchronous designs. On the other hand there are no EDA tools available for asynchronous design implementation. GALS design is a compromise between synchronous system and completely asynchronous system [2]. Each synchronous subsystem (clock domain) can run on its own independent clock frequency. Globally Asynchronous and Locally Synchronous (GALS) technique aims to eliminate the global clock, by partitioning the system into several synchronous blocks and communicating asynchronously among blocks. Globally Synchronous and Locally Asynchronous systems are an intermediate style of design between synchronous and asynchronous designs.

Figure 1. Shows an illustration of Globally Asynchronous and Locally Synchronous (GALS) system [3]. This system can also be called as mixed timing system (or mixed timing network) because the system has mixed-timing interfaces that provides robust communication between the synchronous and asynchronous domains. Moreover, the circuit consists of a set of synchronous terminals with different unrelated clocks and an asynchronous network which is a clock-less network fabric. Here these synchronous modules communicate each other via asynchronous network to provide the low power consumption and electromagnetic interference (EMI) [3].

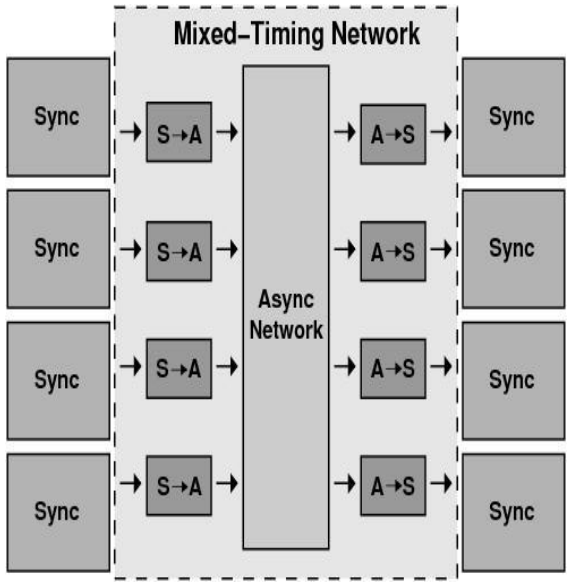


**Figure 1. Globally Asynchronous and Locally Synchronous system [3].**

While implementing GALS architectures the synchronism between blocks will be implemented through four-phase handshaking by the use of request and acknowledge signals. Fig (2a) and Fig (2b) shows 4-Phase hand shaking protocol cycle and transitions which is easy to implement. In this protocol the sender requests the data and sets the request signal to high. Now the received absorbs the data and asserts the acknowledge signal to high. The sender responds by setting request signal to low. Finally received also closes the loop by setting acknowledge signal to low. The other technique is 2-Phase signaling which is level sensitive and difficult to implement. In 2-Phase hand shaking sender sends the data and produce request event. Receive absorbs the data and produces a acknowledgement event.

GALS systems are often highly energy efficient due to their simplified clock tree [4], and their enabling of joint clock and voltage scaling in system sub modules [5], [6]. However, GALS clocking typically also introduces a performance penalty due to additional communication latency between asynchronous domains [5], [7].

Though GALS systems has advantages in terms of dynamic power reduction, leakage power is still an issue down the lower technology nodes. As technologies scale down, percentage of leakage power to total power is gradually going up with every node. Leak-age is an unwanted by-product and substantially reduces the operational time of the devices thereby rendering such devices uncompetitive. It is, therefore, absolutely necessary to eliminate leakage, wherever it is possible [4]. In this paper we are focusing on leakage power elimination along with dynamic power through a new power gating technique. This paper is organized in to six different sections. Section 2 discusses about power gating technique and power gating sequence. In Section 3 we describe the way new proposed power gating sequence using asynchronous wrappers hand shake signals. Section 4 talks about implementation of asynchronous 8051 using GALS architecture and demonstrates the use of handshaking signals for power gating in these architectures for leakage power reduction. Section 5 talks about the results in details. Section 6 talks about conclusions.

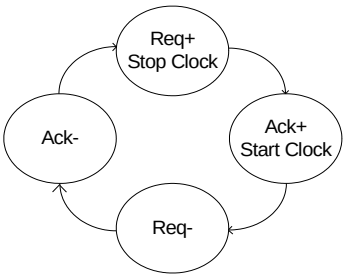


**Figure 2 (a) 4-Phase hand shaking flow diagram**

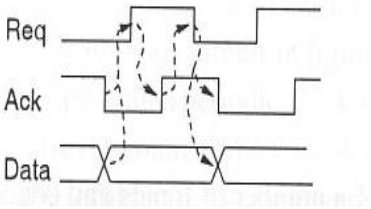


**Figure 2(b). 4-Phase hand shaking signal transitions.**

## 2. POWER GATING TECHINIQUE

Leakage power dissipation grows with every generation of CMOS process technology. This leakage power is not only a serious challenge to battery powered or portable Products. To reduce the overall leakage power of the chip, it is highly desirable to add mechanisms to turn off blocks that are not being used. This technique is known as power gating.

### 2.1 Power Gating

Power gating consists of selectively powering down certain blocks in the chip while keeping other blocks powered up. The goal of power gating is to minimize leakage current by temporarily switching power off to blocks that are not required in the current operating mode.

A simplified view of an SoC that uses internal power gating is shown in Fig (3). Unlike a block that is always powered on, the power-gated block receives its power through a power-switching network. This network switches either VDD or VSS to the power gated block. In this example, VDD is switched; VSS is provided directly to the entire chip. The switching fabric typically consists of a large number of CMOS switches distributed around or within the power gated block.

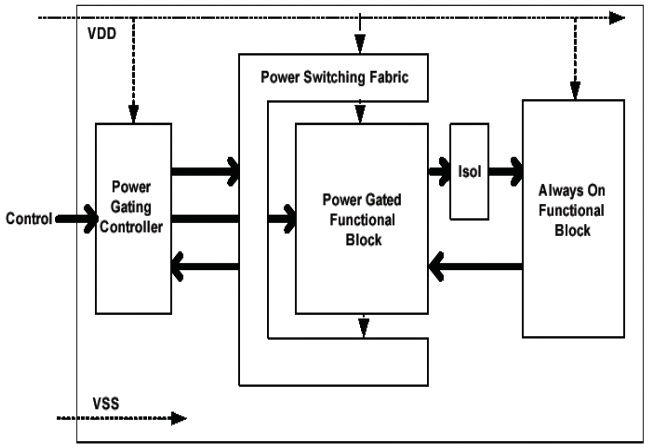


**Figure 3. Power Gating in a SOC [9]**

## 2.2 Power Gating Control

The power gating switch fabric must be designed to limit voltage spikes that might corrupt retention registers or other powered-up logic. Most designs achieve this by limiting the current during power up, and thus limiting the rate at which the voltage rises to its final value. The power controller must accommodate this process. In particular, it must wait until power up is complete before issuing restore [4]. That is, it must insert a delay between power on and restore. A recommend process is to use a request-acknowledge handshake to control the power switching fabric as shown below in Fig (4).

The power controller issues a N_PWR_REQ to turn the power switching fabric off. It is the responsibility of the switching fabric to return N_PWR_ACK when power is completely switched off. On power up, the controller de-asserts N_PWR_REQ to turn the switching fabric on. When the fabric is completely on and it is safe to proceed, the switching fabric de asserts N_PWR_ACK. When the controller sees the acknowledge, it proceeds to assert restore and continue through the power up sequence.

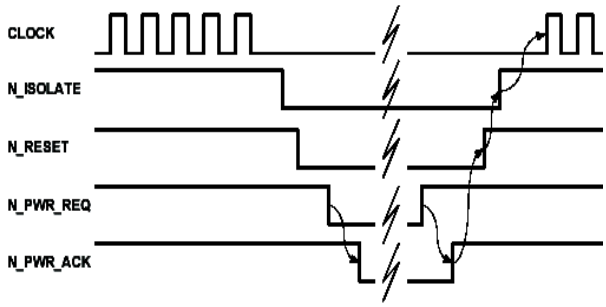


**Figure 4. Power Gating Control [9]**

The key in having similar power gating and power gating sequence in GALS architectures is to use their hand shaking controls of asynchronous wrappers.

# 3. PROPOSED POWER GATING CONTROL FOR GALS ARCHITECTURES

In this work we are proposing a new power gating control which will use existing 4-phase hand shaking "request" signals generated out of asynchronous wrapper of GALS architecture as the control signals. Unlike the power gating control sequence discussed about in our method "request" signal of 4-phase hand shaking control will be used to gate both the clock signal and power.

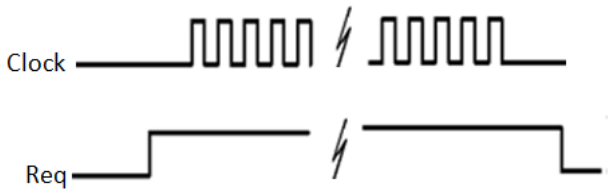


**Figure 5. Request of 4-Phase hand shake signal as power controller**

The advantage of this method is that there is no need for generating external power gating logic. The existing asynchronous wrapper signals can be used to do the power gating. We can plan for Isolation and Retention sequence also by taking extra cycles with respect to these handshaking signals.

# 4. DESIGN ARCHITECTURE

In this work to validate our proposed method we have taken Daltons [1] 8051 synchronous block and created the asynchronous wrappers to meet the GALS criteria. The original synchronous architecture is show in Fig (6).

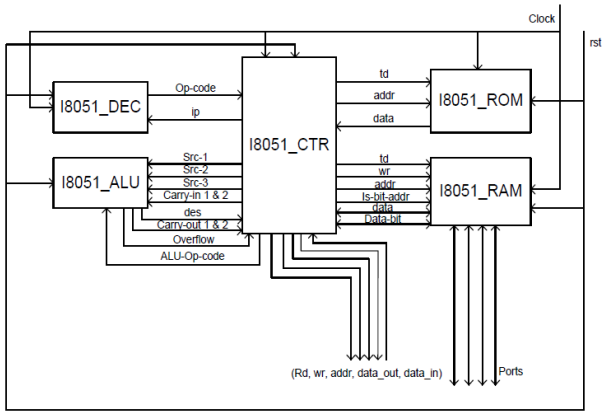


**Figure 6: Block diagram of synchronous 8051 architecture[1].**

Here all the modules are synchronously operated with each other by using a single clock signal called *global clock*. And the controller reads the data from the I8051_ROM module and sends this data to the I8051_DEC module which is a combinational module that decodes that data into an appropriate op-code for the controller to execute the data. Then depending upon the decoded instruction from I8051_DEC module, the I8051_CTR module will assert and deassert the particular control signals to the I8051_ALU and I8051_RAM modules. During the execution phase of a particular instruction, the I8051_CTR module will usually read the data from the I8051_RAM module and sends the accessed data to the I8051_ALU module for the execution of an appropriate logical or arithmetic operation. The results of the ALU operation are written into the I8051_RAM module. To access external hardware, I8051_CTR and I8051_RAM modules feature ports

are used to be interfaced with only the device when the 8051 design is needs to be accessed by an external hardware.

Fig (7) shows the GALS asynchronous architecture with ALU and Controller wrappers. Here the asynchronous 8051 modules does not any have global clock signal as in the synchronous 8051 design. Here the modules are largely asynchronous with each other. Instead of global clock, here the 4-phase handshaking is used to perform the communication between I8051_ALU and I8051_CTR modules with a stoppable clock to block the clock when the controller waits for the ALU to complete a given operation. The operations of the asynchronous microcontroller is similar to the synchronous microcontroller but with a few key differences. Unlike the synchronous version of 8051, here in asynchronous version, the clock is stopped while the I8051_CTR module waits for the I8051_ALU module to execute the result of a given particular operation by using the handshaking signals generated from the ALU and controller wrappers.

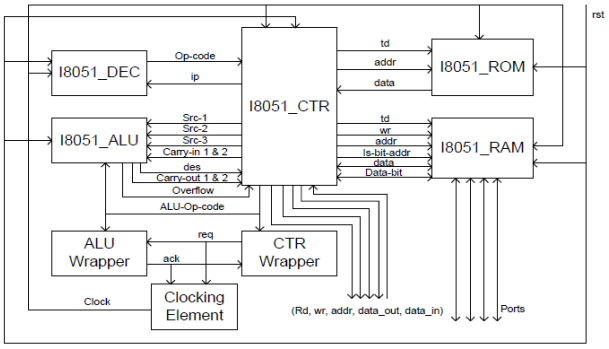


**Figure 7: Block diagram of asynchronous 8051 GALS architecture.**

Since the clock is stopped while the controller waits for the result from the ALU, significant portion of dynamic power will be reduced. In Dalton's[1] model majority of the power consumed by RAM block. Moreover, in synchronous version, clock is generated off the chip, whereas in the asynchronous version, the clock must be generated on board the chip (i.e., stoppable clock).

**Controller Wrapper:** In asynchronous 8051, the controller wrapper produces a *request* signal when the controller needs an operation from ALU. This *request* signal is again deasserted by the controller when it receives an *acknowledge* signal from the ALU wrapper.

**ALU Wrapper:** Here the ALU wrapper produces an *acknowledge* signal to specify that the ALU has completed the given operation which was requested from the controller. This requested operation is determined by the ALU Op-Code. Here the delay time between the ALU wrapper receives the request signal from the controller wrapper and sending the acknowledge signal from the ALU wrapper to the controller wrapper is depends upon the certain logical or arithmetic operation given by the controller.

**Clocking Element:** This unit is used to produce an onboard clock signal for the asynchronous design. The nature of this onboard clock is same as the global clock, except that this clock is stopped when the request signal is asserted and acknowledge signal is deasserted. Thus the excess clock cycles will be blocked with this stoppable clock in the asynchronous design, so that the power consumption will be reduced than in the synchronous design.

Now to address the leakage power we introduced a gating element (power switch) in GALS architecture of 8051 as shown in Fig (8). Fig (8) is power intent diagram for GALS 8051 specified using U.P.F (Unified Power Format) during Synthesis using Synopsys Design Compiler® tool. Here we inserted a power switch in to RAM block which is consuming more power. The idea is reduce its leakage power as well.

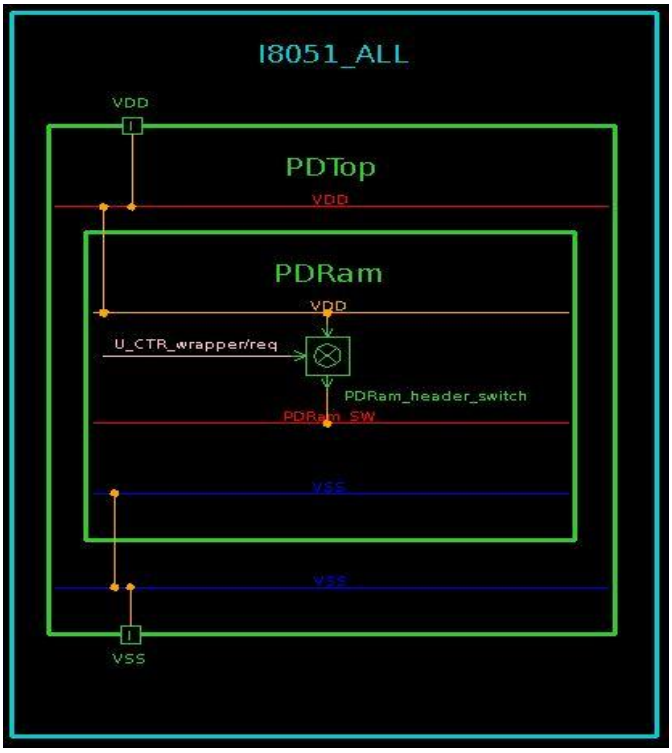


**Figure 8: Power intent specification diagram with power switch for RAM blow which is consuming more power.**

## 5. EXPERIMENTAL RESULTS

The flow diagram show in Fig (9), shows our approach to prove the proposed technique which will reduce leakage power in GALS architectures.

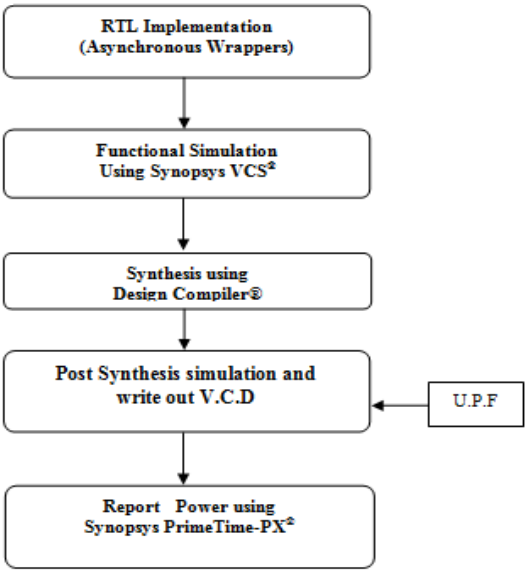


**Figure 9: Design flow used to compare the power reports across synchronous, GALS & proposed methods.**

We carried out pre synthesis simulation for Dalton's synchronous code, GALS version of Daltons code for a division and subtraction operations consequently using Synopsys VCS® by providing equivalent hex code in ROM model. This is because ALU of 8051 takes 140 ns delay for completing a division operation. We have targeted this design for 150 MHz hence CTR block needs to wait for almost 20 cycles during division operation for ALU result. So this is sufficient to prove our proposed methodology by gating a power hungry block. Then we have done the Synthesis using Synopsys Design Compiler® and generated V.C.D (value change dump) files out by using same testbench which is used at simulation stage. After this we used Synopsys PrimeTime-PX® tool to find out the power consumption of synchronous vs. GALS versions of 8051 architectures. Table(1) shows total power consumption of synchronous 8051 design while running given program.

**Table 1. Power Report table of synchronous 8051**

| S.No | Hierarchy | Switching Power | Internal Power | Leakage Power | Total Power | % |
|---|---|---|---|---|---|---|
| 1 | I8051_ALL | 4.78E-05 | 3.15E-04 | 3.58E-04 | 7.21E-04 | 100 |
| 2 | I8051_RAM | 6.15E-06 | 2.05E-04 | 2.08E-04 | 4.19E-04 | 58.2 |
| 3 | I8051_CTR | 3.68E-05 | 1.01E-04 | 1.24E-04 | 2.63E-04 | 36.4 |
| 4 | I8051_ALU | 3.42E-06 | 6.15E-06 | 1.97E-05 | 2.92E-05 | 4.1 |
| 5 | I8051_ROM | 3.38E-07 | 1.83E-06 | 1.02E-06 | 3.19E-06 | 0.4 |
| 6 | I8051_DEC | 1.04E-06 | 5.68E-07 | 4.83E-06 | 6.44E-06 | 0.9 |

**Table 2. Power Report table of GALS 8051**

| S.No | Hierarchy | Switching Power | Internal Power | Leakage Power | Total Power | % |
|---|---|---|---|---|---|---|
| 1 | I8051_ALL | 1.27E-04 | 2.53E-04 | 2.88E-04 | 6.68E-04 | 100 |
| 2 | I8051_RAM | 2.37E-05 | 1.79E-04 | 1.51E-04 | 3.54E-04 | 53 |
| 3 | I8051_CTR | 2.56E-05 | 7.29E-05 | 1.11E-04 | 2.10E-04 | 31.4 |
| 4 | I8051_ALU | 1.96E-06 | 3.38E-06 | 1.95E-05 | 2.49E-05 | 3.7 |
| 5 | I8051_ROM | 2.29E-07 | 1.28E-06 | 1.02E-06 | 2.52E-06 | 0.4 |
| 6 | I8051_DEC | 9.78E-07 | 4.19E-07 | 4.84E-06 | 6.23E-06 | 0.9 |
| 7 | CTR_wrpr | 1.78E-08 | 5.89E-08 | 2.83E-08 | 1.05E-07 | 0 |
| 8 | ALU_wrpr | 1.27E-08 | 5.44E-08 | 2.81E-08 | 9.52E-08 | 0 |
| 9 | clock_gnrtr | 7.45E-05 | N/A | 6.87E-08 | 7.06E-05 | 10.6 |

**Table 3. Power Report table of GALS 8051 with power gating for RAM block to reduce its leakage power**

| S.No | Hierarchy | Switching Power | Internal Power | Leakage Power | Total Power | % |
|---|---|---|---|---|---|---|
| 1 | I8051_ALL | 1.09E-04 | 2.26E-04 | 3.44E-04 | 6.79E-04 | 100 |
| 2 | I8051_RAM | 4.61E-06 | 1.50E-04 | 2.08E-04 | 3.63E-04 | 53.4 |
| 3 | I8051_CTR | 2.62E-05 | 7.34E-05 | 1.11E-04 | 2.10E-04 | 31 |
| 4 | I8051_ALU | 2.49E-06 | 4.54E-06 | 1.98E-05 | 2.68E-05 | 3.9 |
| 5 | I8051_ROM | 2.48E-07 | 1.33E-06 | 1.02E-06 | 2.60E-06 | 0.4 |
| 6 | I8051_DEC | 9.78E-07 | 4.19E-07 | 4.84E-06 | 6.23E-06 | 0.9 |
| 7 | CTR_wrpr | 1.78E-08 | 5.89E-08 | 2.83E-08 | 1.05E-07 | 0 |
| 8 | ALU_wrpr | 1.27E-08 | 5.44E-08 | 2.81E-08 | 9.52E-08 | 0 |
| 9 | clock_gnrtr | 7.45E-05 | N/A | 6.87E-08 | 7.06E-05 | 10.4 |

It also shows the power consumed by RAM, CTR, ALU, ROM and DEC blocks. However please note that power consumed by RAM block is almost 58% of total power. Fig (10) shows the post synthesis results where you can see fb/12 gives 0d. Table (2) shows GALS version of 8051 post synthesis power reports summary. Because of gating clock while ALU is performing division and subtraction there was a 13% reduction in power consumed by RAM block and hence there was a 6% reduction in total power consumed by the GALS 8051 block. 4-Phase hand shaking protocol by request and acknowledge signals of asynchronous wrapper signals can be observed in post synthesis results as shown in Fig (11). Between Table(1) and Table (2) we can observe that there is no change in leakage power consumed by power hungry RAM block. Table (3) shows a reduction 30% reduction in leakage power consumed by RAM block. This is due to power gating specification using U.P.F for RAM block after synthesis and doing the simulation using Synopsys VCS®. Fig (12) shows the CORRUPTED states when the UPF switch is off. While the switch if off RAM block signals will have "x" values dumped in VCD during simulation. After reading the VCD into PrimeTime-PX we observed that there is a reduction of 30% leakage power because of power gating implementation.

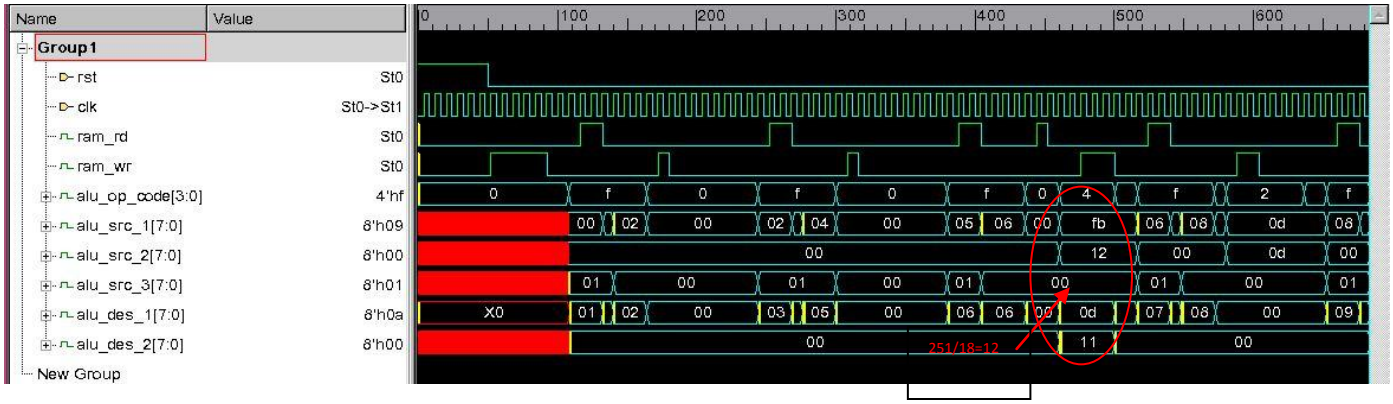


**Figure 10: Post Synthesis simulation results for division operation followed by a substation.**

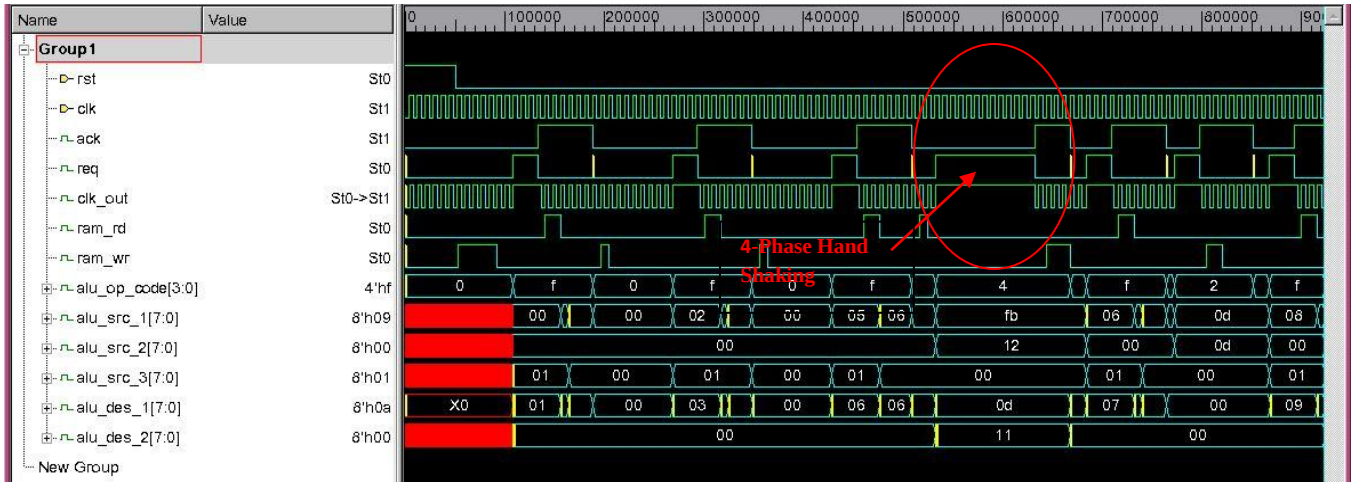


**Figure 11: Post Synthesis simulation results for division operation followed by a substation.**

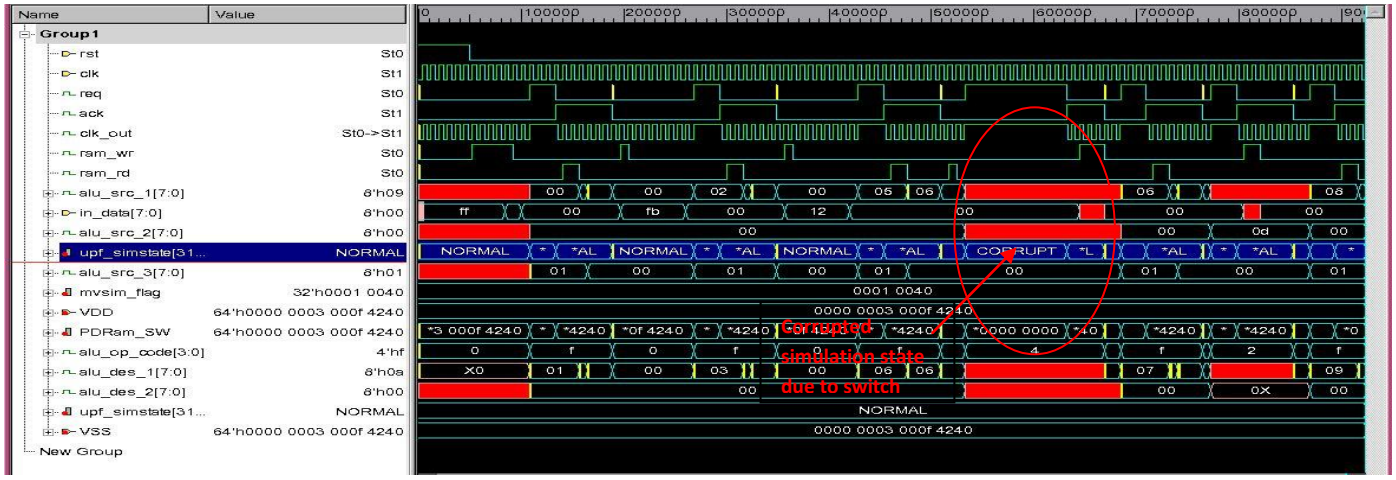


**Figure 12: Post Synthesis simulation results for division operation followed by a substation.**

Power intent specification was done using U.P.F 1.0 version for the GALS 8051 deign.

## 6. CONCLUSIONS

In Deep sub micron nodes there is a need for minimizing leakage power as much as possible. In this paper we proposed a new method to minimize the leakage power of ideal blocks in GALS architecture designs. This approach doesn't require extra circuit over head to build power control logic; instead we can use exiting 4-phase hand shake signals of asynchronous wrapper signals for generating power gating controls. We demonstrated this using U.P.F based post synthesis simulation on a GALS design which was synthesized using Synopsys SAED 90nm library. Apart from this the request and acknowledgement signals of 4-phase hand shaking can also be used as save and restore signal of retention registers which are part of power gating logic. Further to this

## 7. ACKNOWLEDGMENTS

We would like to thank Harish Balan, Sr. Director, Uno V. Nellore, Sr. Manger from Synopsys (India) Pvt. Ltd. for their encouragement and continuous support in doing this research using Synopsys tools

## 8. REFERENCES

[1] Dalton Project.http://www.cs.ucr.edu/~dalton/8051/. University of California, Department of Computer Science, Riverside, CA 92521. 7 April 2005.

[2] Chong-Fatt Law, Bah-Hwee Gwee and Joseph S. Chang, "Modeling and Synthesis of Asynchronous Pipelines", IEEE transactions on Very Large Scale Integrations (VLSI) Systems, Vol 19, No.4.April 2011

[3] Jens Muttersbach, Thoms Villiger, Hubert Kaeslim, Norbert Felber and Wolfgang Fichtner, "Globally-Asynchronous Locally Synchronous Architectures to Simplify the Design of On-Chip System" Proceedings of 12th IEEE international ASIC/SC conference, Washington DC, Sept. 1999.

[4] Michael N. Horak, University of Maryland, Steven M. Nowick, Columbia University, Matthew Carlberg, UC Berkeley Uzi Vishkin, University of Maryland: "Low-Overhead Asynchronous Interconnection Network for GALS Chip- Multiprocessors." IEEE Transactions on April, 2011.

[5] T. Meincke, A. Hemani, S.Kumar, P. Ellervee, J. Oberg, T. Olsson, and P. Nilsson, "Globally asynchronous locally synchronous architecture for large high-performance ASICs," in Proc. IEEE Int. Symp. Circuits Syst., May 1999, pp. 512–515.

[6] G. Semeraro, G. Magklis, R. Balasubramonian, D. H. Albonesi, S. Dwarkadas, and M. L. Scott, "Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling," in Proc. IEEE Int. Symp. High-Perform. Comput. Arch., Feb. 2002, pp. 29–40.

[7] E. Talpes and D. Marculescu, "Toward a multiple clock/voltage island design style for power-aware processors," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 13, no. 5, pp. 591–603, May 2005.

[8] E. Talpes and D. Marculescu, "A critical analysis of application-adaptive multiple clock processor," in *Proc. Int. Symp. Low Power Electron Des., Aug. 2003, pp. 278–281.*

[9] Michael Keating, David Flynn, Robert Aitken, Alan Gibbons and Kaijian Shi."Low Power Methodology anual For System On Chip Designs" 1st ed. 2007. Springer.