Self-Timed SAPTL using the Bundled Data Protocol

K.V.V.Satyanarayana[1] T.Govinda Rao[2] J.Sathish Kumar[3]

[1]Associate Professor
K.L. University
[2, 3]Assistant Professor
Usha Rama college of Engg and technology

Abstract

This paper presents the design and implementation of a low-energy asynchronous logic topology using sense amplifier- based pass transistor logic (SAPTL). The SAPTL structure can realize very low energy computation by using low-leakage pass transistor networks at low supply voltages. The introduction of asynchronous operation in SAPTL further improves energy-delay performance without a significant increase in hardware complexity. We show two different self-timed approaches: 1) the bundled data and 2) the dual-rail handshaking protocol. The proposed self-timed SAPTL architectures provide robust and efficient asynchronous computation using a glitch-free protocol to avoid possible dynamic timing hazards. Simulation and measurement results show that the self-timed SAPTL with dual-rail protocol exhibits energy-delay characteristics better than synchronous and bundled data self-timed approaches in 180-nm, 120-nm CMOS.

Keywords: pass transistor, self-timing, sense amplifier-based pass transistor logic (SAPTL)

I. Introduction

A CMOS technology continues to scale, both supply Voltage and device threshold voltage must scale down Together to achieve the required performance. Lowering the supply voltage effectively reduces dynamic energy consumption but is accompanied by a dramatic increase in leakage energy due to the lower device threshold voltage needed to maintain performance [1].As a result, for low-energy applications, the leakage energy that the system can tolerate ultimately limits the minimum device threshold voltage. Speed, therefore, benefits little from technology scaling. The sense amplifier-based pass transistor logic (SAPTL) [2] is a novel circuit topology that breaks this tradeoff in order to achieve very low energy without sacrificing speed. The initial SAPTL circuits were designed to operate synchronously [2] but with the intent of being able to Operate asynchronously with some minor modifications.

As the effects of process variations continue to increase dramatically with technology scaling, it is becoming harder to design variation-tolerant timing schemes using the traditional synchronous methodologies. To meet a certain timing requirement, the synchronous approach must use a very conservative "worst case" design that is slow enough for the needs of the statistically slowest circuit elements and, thus, will fail to exercise the whole capacity of statistically faster parts of the circuit. The asynchronous approach, on the other hand, can exploit local timing information to achieve "average-case" performance. An asynchronous design can get the best performance out of all components independent of statistical variations in local speed while guaranteeing correct circuit operation.

Asynchronous operation is also attractive to the low-power designer. The absence of a clock distribution network can significantly reduce the power overhead needed to generate timing information. Furthermore, an idle asynchronous system avoids consuming any active power. Despite the advantages of asynchronous operation, the circuit complexity and performance overhead required to implement the needed handshaking protocol may not be trivial. The overhead cost might offset all benefits and make the asynchronous approach impractical. The SAPTL, however, offers a relatively easy way to realize asynchronous operation. Because of the differential signaling used, it is easy to determine when a logical operation completes. Therefore, the self-timed SAPTL topology is a promising candidate for reducing power consumption and improving speed in extremely low energy applications.

II. Saptl Architecture

The basic architecture of the SAPTL circuit is shown in Fig. 1. It is composed of a pass transistor stack, a driver, and a sense amplifier [2]. The SAPTL achieves low energy operation 1) by decoupling sub threshold leakage current from the stack threshold voltage, allowing for increased performance without an increase in leakage energy, and 2) by confining sub threshold leakage to well-defined and controllable paths found only in the drivers and sense amplifiers.
Note that the total energy consumed by the SAPTL is composed of the following: 1) the energy used by the driver to energize the stack; 2) the energy used by the sense amplifier to resolve the correct logical levels and drive the inputs of the fan-out stacks; and (3) the energy needed to generate the appropriate timing information, either globally, such as clock distribution networks, or locally, as in handshaking circuits.

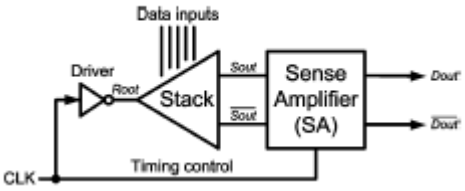

Fig.1 Architecture of SAPTL module with synchronous timing control.

A. Stack and Driver

The stack consists of an NMOS-only pass transistor tree with full-swing inputs and low-swing pseudo differential outputs to perform the required logic function, as shown in Fig. 2. The stack can implement any Boolean expression by connecting the min term branches of the tree to one output and the max term branches to the other as illustrated by the programming switches in the diagram. In our current implementation, the logic function of an SAPTL stack is determined and permanently fixed at fabrication by replacing the programming switches with hardwired connections. Because the stack has no supply rail connections, it does not contribute sub threshold leakage current, and it also has no gain.

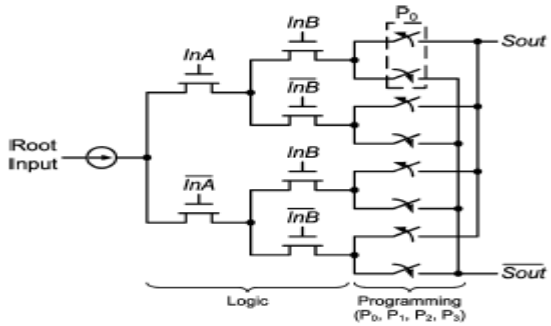

Fig.2 Schematic of a two-input stack with Nstack =2.

A driver, which is a simple inverter in this case, injects an evaluation current into the root of the stack. In operation, either Sout or Sout bar , but not both, is charged toward the supply rail when the driver energizes the selected path through the stack. After each computation and before every evaluation, both differential outputs are reset to ground (logical "0") to initialize the stack to a known state. This initialization is done by turning on all the transistors in the stack and draining the charges out through the root of the stack when the driver output is zero. The alternate charging and resetting of Sout or Sout bar realizes a standard dual-rail encoding scheme [3].

The speed of the SAPTL module depends strongly on the depth of the stack Nstack, which is defined as the number of transistors in series from the root node to the differential outputs. Because the stack contributes no sub threshold leakage current, the stack transistors can have a very low threshold voltage and still operate in the super threshold region even with a very low supply voltage. Therefore, SAPTL is a promising candidate to realize ultralow energy computation without entering the sub threshold region of operation [2].

B. Sense Amplifier

The sense amplifier, shown in Fig. 3, serves three purposes: 1) it amplifies the low-voltage stack output, restoring the signal to full voltage; 2) it serves as a buffer stage at the output of the stack, so as to improve overall speed; and 3) it pre charges both its outputs to Vdd (logical "1"), allowing the reset of the driven fan-out stacks. The sense amplifier consists of two stages. The first stage acts as a preamplifier to reduce the impact of mismatch in the actual technology environment, and the second stage acts as a cross-coupled latch which retains the processed data even after the stack is reset. The sense amplifier is designed to detect input voltages that are less than $(V_{dd} - V_{th})$, thus reducing the performance degradation due to the low stack voltage swings and the absence of gain in the pass transistor network. By turning off the driver as soon as the sense amplifier makes a decision, the stack voltage swings are kept to a minimum, reducing the energy required to perform the desired logical operation.

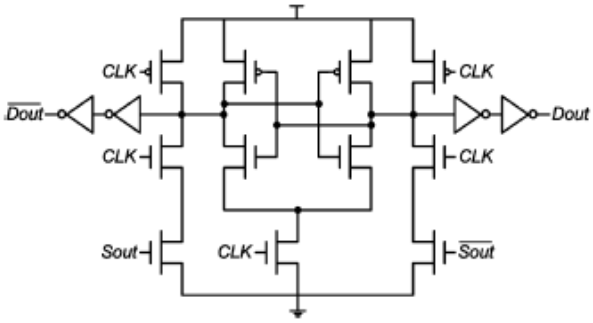

Fig.3 Sense amplifier circuit.

The leakage of the sense amplifier accounts for most of the leakage energy of the SAPTL module. It can be directly traded off against the input sensitivity of the sense amplifier to size and threshold voltage mismatch as shown in Fig. 4. Using a supply voltage of 300 mV and a minimum input voltage of 100 mV, 55% of the sense amplifier leakage is due to the four output buffers (inverters). Thus, an increase in sense amplifier performance can be achieved 1) by reducing the minimum input voltage or 2) by increasing the output drive of the sense amplifier, either of which would result in an increase in leakage current.

III. Bundled Data Self-Timed Saptl Design

The circuit implementation of the self-timed SAPTL module using the bundled data protocol [7] is shown in Fig. 4. The main data path, composed of a driver and stack, evaluates data or resets after receiving the request signal Rqein and data input signals Din and Din bar from the previous SAPTL stage. The control path, which consists of a delay line and a C-element, produces the local clock signal Enable to trigger the sense amplifier.

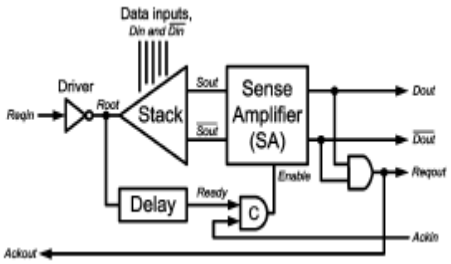

Fig.4 Architecture of self-timed SAPTL module with bundled data protocol.

The delay line mimics the delay of the stack to generate the control signal Ready indicating that the stack has finished an operation. The C-element then produces Enable by collecting Ready and the acknowledge signal Ackin from the next SAPTL stage. In multiple fan-in and fan-out situations, additional C-element can be employed to reconverge multiple request and acknowledge events from the different fan-in and fan-out stages. When triggered by Enable , the sense amplifier latches the stack output data or resets depending on the logical state of Enable. The full-swing data output signals Dout and Dout bar are made available at the outputs of the sense amplifier. The AND gate serves

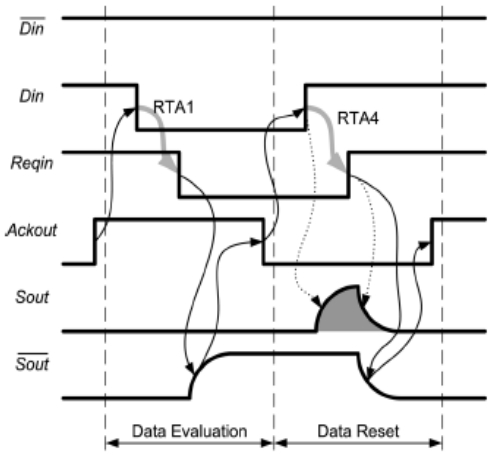

Fig.5 Timing diagram of self-timed SAPTL**.**

As a completion detection circuit, generating the handshake signals Ackout and Requot that indicate the completion of the current operation. We can summarize the relationship between the input and output signals of the th SAPTL stage as

$$\begin{aligned} \mathrm{Dout}_i, \overline{\mathrm{Dout}_i} &= f\left(Enable_i, \mathrm{Sout}_i, \overline{\mathrm{Sout}_i}\right) \\ &= ff\left(Reqin_i, Ackin_i, Din_i, \overline{Din_i}\right) \quad (1) \\ Ackout_i &= Reqout_i \\ &= g\left(\mathrm{Dout}_i, \overline{\mathrm{Dout}_i}\right) \quad (2) \\ Enable_i &= h\left(Reqin_i, Ackin_i\right) \quad (3) \end{aligned}$$

where for the subsequent $(i+1)$th SAPTL stage

$$\begin{aligned} Din_{i+1} &= \mathrm{Dout}_i \\ \overline{Din_{i+1}} &= \overline{\mathrm{Dout}_i} \\ Reqin_{i+1} &= Reqout_i \\ Ackout_{i+1} &= Ackin_i. \end{aligned}$$

IV. Dual-Railself-Timed Saptl Design

In a self-timed SAPTL structure using the bundled data protocol, RTA2 is the most critical design constraint. In order to guarantee correct operation under process, voltage, and temperature variations, the latency of the delay line can become very large and can severely limit the overall performance. Because the SAPTL uses dual-rail coding to represent data, we can use the output signals of the stack and, instead of from the delay line, to trigger the C-element. As a result, we can 1) eliminate the delay line and 2) design the C-element to respond immediately after the stack finishes operation, without being limited by RTA2.

Furthermore, we can combine the sense amplifier and C-element circuits into a composite block through gate-level optimization, yielding a more energy-efficient architecture, as shown in Fig.7. The optimized architecture with dual-rail protocol [7] eliminates the traditional

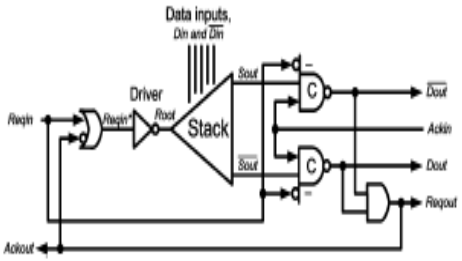

Fig.6 Architecture of glitch-free self-timed SAPTL module with dual-rail protocol.

Sense amplifier circuit and directly employs two C-element circuits as a complex gain stage at the outputs of the stack. The overall conversion speed, however, may be slower than the design with a sense amplifier due to the absence of a differential amplification and

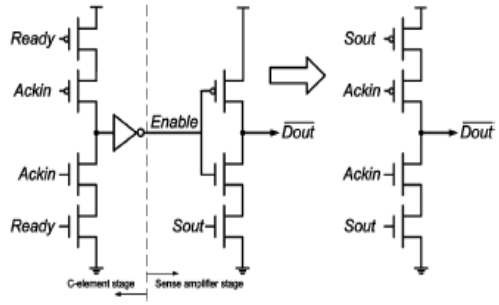

Fig.7 Logic combination of two-input C element and sense amplifier circuits.

The loss the loss of a positive-feedback mechanism between the two data paths Fig.6 shows the implementation of a glitch-free self-timed SAPTL architecture without the delay line.

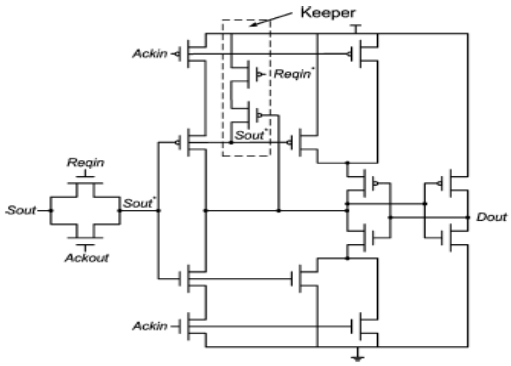

Fig.7: Two-input C-element circuit with additional decision-making logic for glitch-free self-timed SAPTL.

The design and performance of the C-element circuits are particularly important in this architecture because the C-element not only plays the role of the gain stage but also serves as the handshaking element. The self-timed SAPTL with dual-rail protocol has latency and cycle time expressions similar to (4)–(8). Note that the speed enhancement discussed in Section IV-C does not apply to the dual-rail design in Fig.6, because of the absence of the internal signal. However, the single self-timed SAPTL stage is now elastic and able to achieve the best performance across process, voltage, and temperature variations and different input characteristics. The self-timed SAPTL can thus exercise the full potential of asynchronous computation without the limitations of the delay line. It is interesting tonote that the optimized self-timed SAPTL architecture in Fig.6.

Has almost the same hardware complexity as the original synchronous SAPTL design in Fig. 1. This means that, with almost "zero cost," SAPTL is able to achieve both better performance and better robustness in the presence of variability by operating asynchronously.

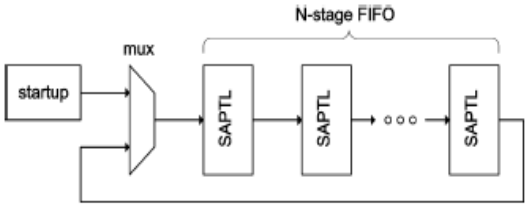

Fig.8 Test setup for energy and delay measurements.

The energy and delay of the various SAPTL5
Implementations were measured using N=8**.**

V. Performance Evaluation And Comparison

We evaluated and compared the performance of the self-timed and synchronous SAPTL circuits using the Spectre circuit simulator. We also performed Monte Carlo simulations to ensure the correct operation of the SAPTL circuits even with 6σ process variations. In addition, we implemented the self-timed SAPTL circuits in a 120-nm CMOS test chip and compared the actual measurement results to the simulated data. This section presents the energy-delay and leakage comparisons of synchronous versus self-timed SAPTL. The simulations exclude the parasitic contributions from the interconnect wires and the clock network. However, the effect of global parameters, such as clocks and long interconnect wires, should be done at the system level, in the context of an actual application. Comparisons between the synchronous SAPTL and other logic styles can be found in [2].

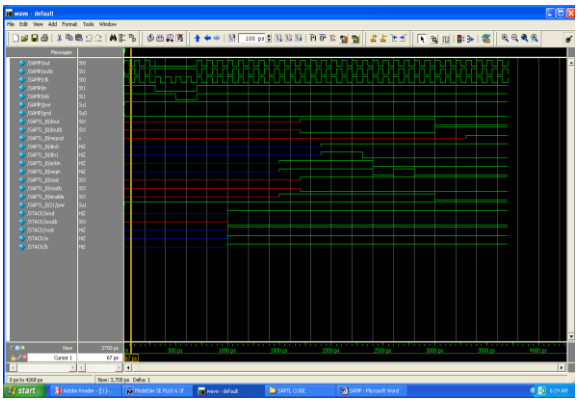

Fig .9.1 Simulation results

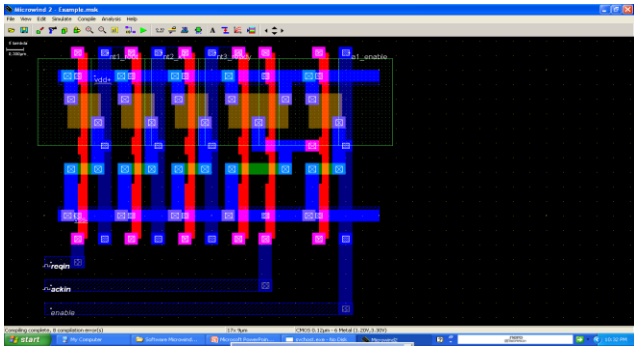

Fig .9.2 Layout design for Bundled –Data Self-Timed SAPTL

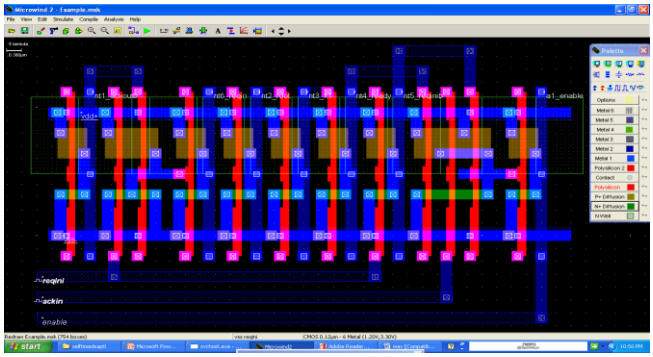

Fig .9.3 Layout design for Dual-rail-self timed SAPTL

Energy-Delay Characteristics

The pre-layout simulated energy and delay behavior of the synchronous SAPTL5 and both versions of the self-timed SAPTL5 is shown in Fig.9.4.

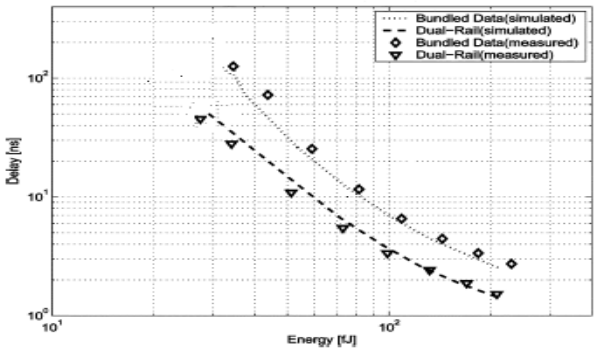

Fig.9.4 Measured versus simulated energy-delay plots for The 120-nm CMOS Self-timed SAPTL5, as the supply Voltage is varied from 300 mV to 1.2 V

echnology	**ower 80 nm Technology)**	**Power 20 nm echnology)**
ynchronous	6.87µW	6.58µW
undled-data	6.35µW	6.64µW
ual-rail	6.64µW	99µW

Table .1 Comparisons of different Methodologies

V. Conclusion

The asynchronous operation of the SAPTL provides robustness in the presence of variability as well as performance advantages over synchronous operation. While the self-timed SAPTL using the bundled data protocol can potentially achieve higher speed performance by overlapping the data evaluation and reset cycle, the self-timed design based on the dual-rail protocol has less rigid relative timing constraints, which leads to better energy and speed performance in technologies with increased process variations. The early reset operation ofself-timed SAPTL not only prevents dynamic timing hazards from glitches but also improves both energy and speed performance. We evaluated and compared the performance of the self-timed and synchronous SAPTL circuits using the Spectre circuit simulator. We also performed Monte Carlo simulations to ensure the correct operation of the SAPTL circuits even with process variations. In addition, we implemented the self-timed SAPTL circuits a 120-nm CMOS test chip and compared the actual measurement results is less than the simulated data. And also 180-nm CMOS is less than the simulated data This section presents the energy-delay and leakage comparisons of synchronous versus self-timed SAPTL. The simulations exclude the parasitic contributions from the interconnect wires and the clock network. However, as pointed out earlier in Section II-D, the effect of global parameters, such as clocks and long interconnect wires, should be done at the system level, in the context of an actual application. Comparisons between the synchronous SAPTL and other logic styles can be found in [2].

REFERENCES

[1] T. Sakurai, “Perspectives on power-awar electronics,” in ISSCC Dig. Te ch. Papers, 2003, vol.1, pp. 26–29.

[2] L. Alarcón, T.-T. Liu, M. Pierson, and J. Rabaey, “Exploring very lowenergy logic: A case study,” J. Low Power Electron., vol. 3, no. 3, pp. 223–233, Dec. 2007.

[3] J. Sparsø and S. Furber, Principles of Asynchronous Circuit Design.Norwell, MA: Kluwer, 2001.

[4] J. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits:A Design Perspective, 2nd ed . Englewood Cliffs, NJ: Prentice-Hall, 2003.

[5] H. Li, S. Bhunia, Y. Chen, K. Roy, and T. Vijaykumar, “DCG: deterministic clock-gating for low-power microprocessor design,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 12, no. 3, pp. 245–254,Mar. 2004.

[6] N. Banerjee, K. Roy, H. Mahmoodi, and S. Bhunia, "Low power synthesis of dynamic logic circuits using fine-grained clock gating," in Proc. DATE, Mar.2006, vol. 1, pp. 1–6.

[7] T.-T. Liu, L. Alarcón, M. Pierson, and J. Rabaey, "Asynchronous computing in sense amplifier-based pass transistor logic," in Proc. 14th IEEE Int. Symp. ASYNC, Apr. 2008, pp. 105–115.

[8] T. Williams, "Performance of iterative computation in self-timed rings," J. VLSI Signal Process., vol. 7, no. 1/2, pp. 17–31, Feb. 1994.

[9] K. Stevens, R. Ginosar, and S. Rotem, "Relative timing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 1, pp. 129–140, Feb. 2003.

[10] I. Sutherland, "Micropipelines," Commun. ACM, vol. 32, no. 6, pp.720–738, Jun.1989.

[11] S. Narendra, "Scaling of stack effect and its application for leakage reduction," in Proc. ISLPED, Aug. 2001, pp. 195–200.

[12] K. Yano, Y. Sasaki, K. Rikino, and K. Seki, Top- down pass-transistor logic design. IEEE Journal of Solid-State Circuits 31, 792 (1996).

13] R. Shelar and S. Sapatnekar, BDD decomposition for delay oriented pass transistor logic synthesis. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 13, 957 (2005).

[14] P. Buch, A. Narayan, A. Newton, and A. Sangiovanni-Vincentelli, Logic synthesis for large pass transistor circuits. IEEE/ACM International Conference on Computer-Aided Design, November (1997), pp. 663–670.

[15] W. C. Elmore, The transient analysis of damped linear networks with particular regard to wideband amplifiers. J. Appl. Phys. 19, 55 (1948).

[16] V. Kheterpal, V. Rovner, T. Hersan, D. Motiani, Y. Takegawa, A. Strojwas, and L. Pileggi, Design methodology for IC manufacturability based on regular logic-bricks. Proceedings of the 42nd Design Automation Conference, June (2005), pp. 353–358.

K. V. V. Satyanarayana received the M.Tech degree from Jawaharlal Nehru Technological University, Kakinada in 2008 in Electronics and Communication engineering. He is an Associate Professor in the Department of Electronics and Communication engineering, K.L.University,Vaddeswaram. His current research interests include the area of Communications, video coding techniques, and Architectures design.

T. GOVINDA RAO received the M.Tech (VLSISD) degree from Jawaharlal Nehru Technological University, Kakinada in 2011 in Electronics and Communication engineering. he is an Assistant Professor in the Department of Electronics and Communication engineering, Usha Rama College Of Engineering and Technology, Vijayawada. His current research interests include the areas of very large scale integration (VLSI) testing and fault-tolerant computing, video coding techniques, and Architectures design.

J. SATHISH KUMAR received the M.Tech degree from Jawaharlal Nehru Technological University, Hyderabad in 2011 in Electronics and Communication Engineering. He is an Assistant Professor in the Department of Electronics and Communication Engineering, Usha Rama College of Engineering and Technology, Vijayawada. His current research interests include the areas of very large scale integration (VLSI) testing and fault-tolerant computing, video coding techniques and Architectures design.