ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 4, April 2018

DSP Accelerator Architecture Using Carry-Save Arithmetic

[1]E. Ramya, [2] Roja. S, [3] Roja.R, [4] Sandhiya.G
[1][2] [3] Sns college of technology (an autonomous institution)

Abstract: **- A high performance data path to implement DSP kernels are introduced in this paper .Hardware acceleration has been proved an extremely promising implementation DSP domain. We present a novel accelerator architecture comprising flexible computational units for the execution of a large set of operation templates in DSP kernels. Carry Save method has been implemented to improve the performance of the accelerator while computing more bits. Advanced arithmetic design concepts, i.e., recoding techniques, are utilized enabling CS optimizations to be performed in a larger scope than in previous approaches. Accelerator architecture delivers average gain compared with the state-of-art flexible datapaths.**

KEYWORDS**- Arithmetic optimizations, carry-save (CS) form, datapath synthesis, flexible accelerator, flexible datapath operation chaining, recoding techniques.**

I. INTRODUCTION

Latest embedded systems aim at high speed performance and solving the complexity of digital signal processing (DSP) functions through specialized hardware accelerators [1]. Though application-specific integrated circuits (ASICs) are ideal acceleration solution for performance and power, they also leads to increased silicon complexity, as complex ASICs are needed to accelerate various kernels. Many researchers had found to increase ASICs' flexibility without significantly compromising their performance. Flexible datapaths [2], [4], [6], [7], [10] are used to flow graph (DFG) of a kernel. Design decisions have high impact its efficiency. Existing works mainly exploit architecture-level optimizations, e.g., increased instruction-level parallelism (ILP) [2]–[5], [7] and operation chaining. The domain-specific architecture of [5] and [9] efficiently map primitive or chained operations in the initial data-vary the type and number of computation units for a customized design structure. Recently, aggressive operation chaining have been implemented for the computation of entire subexpressions using multiple ALUs with various arithmetic features. Manuscript received July 25, 2014; revised October 16, 2014 and December 12, 2014; accepted January 8, 2015.

The authors are with the Department of Electrical and Computer Engineering, National Technical University of Athens, Athens 106 82, Greece (e-mail: kostastsoumanis@microlab.ntua.gr;sxydis@microlab.ntua.gr; zervakis@microlab.ntua.gr; pekmes@microlab.ntua.gr).
Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org. Digital Object Identifier 10.1109/TVLSI.2015.2390974

The above described architectures exclude arithmetic optimizations during the architectural synthesis and consider them only at the internal circuit structure of primitive components, e.g., adders, during the logic synthesis [11].The [12]–[14] arithmetic optimizations at higher abstraction levels than the structural circuit have significant impact on the datapath performance. In [12], timing-driven optimizations based on carry-save (CS) arithmetic were performed at the post-Register Transfer Level (RTL) design stage.Verma et al. [14] developed modifications on DFG to maximize the use of CS arithmetic. The aforementioned CS optimization approaches target inflexible datapath, i.e., ASIC, implementations. Recently, Xydis et al. [6], [7] proposed a flexible architecture combining the ILP and pipelining techniques with the CS-aware operation chaining. However, all the above solutions feature a strong drawback, i.e., CS optimization is bounded to merge only additions/subtractions. A CS to binary conversion is inserted before each operation which differ from addition/subtraction, e.g., allocating multiple CS to binary conversions heavily degrades performance due to time-consumptions in carry propagations. In this paper, we propose a high-performance architectural scheme for the generation of flexible DSP accelerators by grouping optimization techniques from both the architecture and arithmetic levels of abstraction. We propose a flexible datapath architecture that exploits CS optimized templates of chained operations that comprises of flexible computational units (FCUs). It delivers average gain up to 61.91% in area-delay product and 54.43% in energy

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 4, April 2018

consumption compared to state-of-art flexible datapaths [4], [7], managing efficiency toward scaled technologies.

II. CARRY-SAV E AR I TH M ETI C : MOT IVAT IONAL OBSE RVAT I ONS AND DRAWBACKS

CS concepts [15] has been widely used to construct arithmetic circuits having high performance and to eliminate carry-propagation chains. They have multiple input additive operations (i.e., chained additions in the initial DFG), mapping onto CS compressors. The goal is to widen the CS computation within the DFG. A multiplication node is inserted in the DFG, either from CS to binary conversion [12] or the DFG is transformed using the distributive property [14]. CS optimization have limited impact on DFGs e.g., filtering DSP applications. In this brief, we solve the limitation by making use of the CS to modified Booth (MB) [15] and recoding each time a multiplication within a CS-optimized datapath. Thus, the

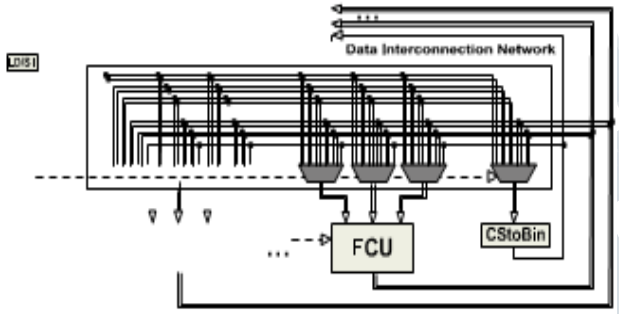


Fig. 1. Abstract form of the flexible datapath.

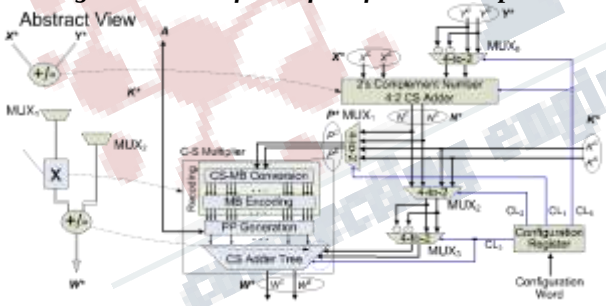


Fig. 2. FCU.

computations are processed using CS arithmetic. Without using any intermediate carry-propagate adder the operation in the targeted datapath are carried in CS to binary conversion,for enhancing the performance.

III. PRO POSE D FLEX I B LE ACC ELER ATO R

The flexible accelerator architecture is shown in Fig. 1. Each FCU operates 16 bit operand directly on CS operands and produces output in the same form1 for reuse of results obtained. Such a bit-length is adequate for the most DSP datapaths [16]. But the architectural concept of the FCU are also adapted for smaller or larger bit-lengths. The number of FCUs arebased on the ILP and area constraints. The CStoBin module is a ripple-carry adder and converts the CS form to the two's complement one. The register bank consists of scratch registers, used for storing intermediate results and sharing operands among the FCUs. Different DSP kernels (i.e., different register allocation and data communication patterns per kernel) can be mapped onto the proposed architecture using post-RTL datapath interconnection sharing techniques [9], [17], [18]. The control unit drives the overall architecture (i.e., communication between the data port and the register bank, configuration words of the FCUs and selection signals for the multiplexers) in each clock cycle.

A. Structure of the Proposed Flexible Computational Unit

The structure of the FCU (Fig. 2) designed are used for high-performance flexible operation chaining based [4], [7]. Each FCU can be configured to any of the T1–T5 operation templates shown in Fig. 3. The FCU also enables intratemplate operation chaining by fusing the additions

1 The FCU operates on either CS or two's complement formatted operands

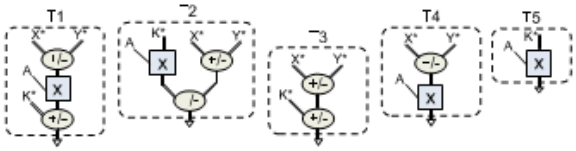


Fig. 3. FCU template library.

performed before/after the multiplication and performs any partial operation template of the following complex operations:

$$W^* = A \times (X^* + Y^*) + K^* \quad (1)$$

$$W^* = A \times K^* + (X^* + Y^*). \quad (2)$$

The following relation holds good for all CS data:

$$X^* = \{X^C, X^S\} = X^C + X^S.$$

The operand A is a two's complement number. The control signals of the multiplexers MUX_1 and MUX_2 (Fig. 2) are set for the alternative execution paths in each FCU. The output of MUX_0 Y^* when $CL_0 = 0$ (i.e., $X^* + Y^*$ is carried out) or Y^* when $X^* - Y^*$ is required and $CL_0 = 1$. The

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 4, April 2018

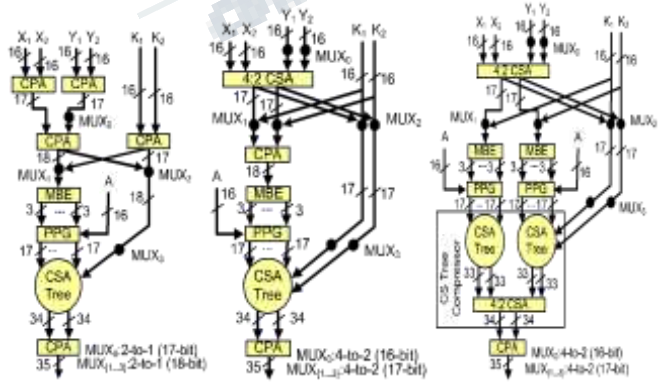
two's complement 4:2 CS adder produces the output $N^* = X^* + Y^*$ when the input carry equals 0 or the $N^* = X^* - Y^*$ when the input carry equals 1. The MUX_1 specifies if N^* (1) or K^* (2) is multiplied with A. The MUX_2 specifies if K^* (1) or N^* (2) is added with the product. The multiplexer MUX_3 consists of the output from MUX_2 and its 1's complement and outputs the former one when an addition with the multiplication product is required (i.e., $CL_3 = 0$) or the later one when a subtraction is carried out (i.e., $CL_3 = 1$). The 1-bit ace for the subtraction is added in the CS adder tree. The multiplier consists of CS-to-MB module, [19] to recode the 17-bit P^* in its MB digits with minimal carry propagation. The product consists of 17 bits. The compensation method is included to eliminate the error during truncation [20].

B. DFG Mapping Onto the Proposed FCU-Based Architecture

In order to map DSP kernels onto the proposed FCU-based accelerator, the semiautomatic synthesis methodology [7] has been adapted. Firstly, a CS operation is performed onto the original DFG, merging nodes of multiple chained additions/subtractions to 4:2 compressors to form FCU template operations (Fig. 3). The designeropt for the FCU operations having the DFG with minimized delay.

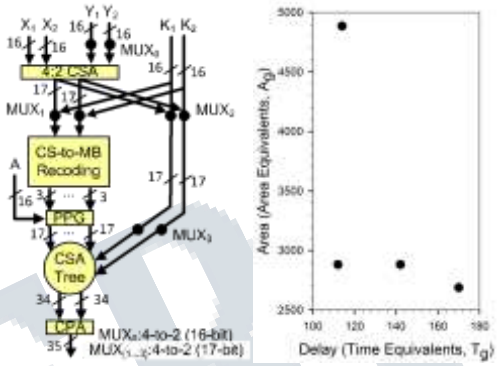
Number of FCUs is fixed, a resource-constrained scheduling is considered with the available FCUs and CStoBin modules determine the resource constraint set. The clustered DFG is assigned to a specific control step. A list-based scheduler [21] has been adopted considering the mobility2 of FCU operations according to descending mobility. These operations are bound onto FCU instances and proper bits are generated. After completing register allocation, a FSM is generated to implement the control unit of the overall architecture.

2 Mobility: The ALAP-ASAP difference of the FCU operations.



(a) *(b)* *(c)*

Fig. 4. Typical chaining of addition–multiplication–addition operations reflecting T1 template of Fig. 3. Its design is based on (a) two's complement arithmetic, (b) CS optimizations of [12], (c) CS optimizations with multiplication distribution [14], and (d) incorporating the CS-to-MB recoding concept. (e) Positioning of the proposed approach with respect to the two's complement one and the CS optimizations based on [12] and [14].



(d)(b)

IV. THEORE T I CAL ANA LY S I S

Based on the unit gate model, we provide the theoretical analysis of the proposed approach. Fig. 4(a) shows the AMADFG when all operands are in two's complement form. Fig. 4(b) shows CS optimizations of [12] . Fig. 4(c) illustrates how [14] distributes the multiplication operation over the CS formatted data. The Fig. 4(d) incorporates the CS-to-MB recoding unit. We assume 16-bit input operands for all the designs and, without loss of generality, we do not consider any truncation concept during the multiplications. Fig. 4(e) shows the positioning of the proposed approach based on [12] and [14].The proposed design solution is the most effective among all the design alternatives.

V. EX P ER I M EN TA L EVA L UAT I O N

A .Proposed FCUs circuit level exploration

A circuit-level comparative study was conducted among the proposed FCU, the flexible computational component (FCC) of [4] and the reconfigurable arithmetic unit (RAU)4 of [7] in scaled technology nodes.bits of the respective two's complement form in scaled .The scaling impact on the performance does not eliminate the benefits of using CS arithmetic. The three units considered were described in RTL using Verilog code .The CSA tree and the adders and multipliers of the FCC were imported from the Synopsys Design Ware library [11]. We used Synopsys Design Compiler [11] to synthesize the examined units and the TSMC 130, 90, and 65nm

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 4, April 2018

standard cell libraries.5 Each unit is synthesized with the highest optimization degree at its critical clock period and 20 higher ones with a step interval of 0.10 ns. Fig. 5 reports the area complexity of the evaluated units

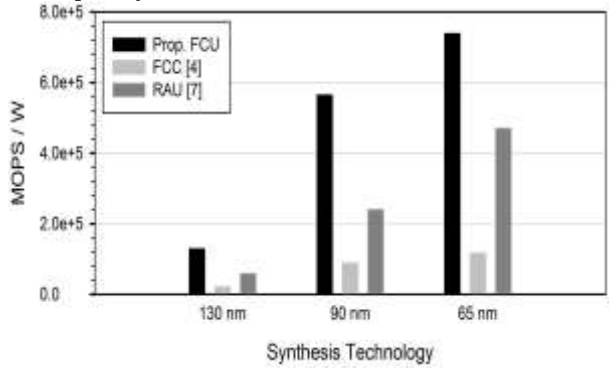


Fig. 6. MOPS/W values of FCUs at the lowest achievable clock periods with respect to the synthesis technology.

FCC, and the RAU operate without timing violations starting at 2.98, 4.83, and 1.99 ns, respectively. At 90 nm, the proposed FCU, the FCC, and the RAU are timing functional starting at 1.66, 2.46, and 1.01 ns, respectively .At 65 nm, the proposed FCU, the FCC, and the RAU start operating without timing violations at 1.13, 1.68, and 0.67 ns, respectively .Fig. 5 shows that the proposed FCU outperforms the FCC in terms of critical delay and area complexity, but presents larger values for these metrics than the RAU in all the technology nodes. However, RAU's flexible pipeline stage (FPS) [7] features limited ability in carrying out heavy arithmetic operations as shown from the mega operations per second/watt (MOPS/W)6 evaluation in Fig. 6. Fig.6 shows the MOPS/W values for the proposed FCU, the FCC, and the RAU at their critical clock periods with respect to the synthesis technology. For each unit, we consider the templates

VI. CONCL U SION

In this detailed description, we have proposed a flexible accelerator architecture that the incorporates CS arithmetic operations for fast switching of addition and multiplication operations. Thus the flexible accelerator is able to operate on both conventional two's complement and CS-formatted data, enabling high degree of computation. Theoretical and practical analyses have shown that the above solution provides an effective design tradeoff point with optimized latency/area and energy implementations. This accelerator enhances the performance than anyother,providing better reliability.

REF ER EN C E S

1. P. Ienne and R. Leupers, Customizable Embedded Processors: Design Technologies and Applications. San Francisco, CA, USA: Morgan Kaufmann, 2007.

2. P. M. Heysters, G. J. M. Smit, and E. Molenkamp, "A flexible and energy-efficient coarse-grained reconfigurable architecture for mobile systems," J. Supercomput., vol. 26, no. 3, pp. 283–308, 2003.

3. B. Mei, S. Vernalde, D. Verkest, H. D. Man, and R. Lauwereins, "ADRES: An architecture with tightly coupled VLIW processor and coarse-grained reconfigurable matrix," in Proc. 13th Int. Conf. Field Program. Logic Appl., vol. 2778. 2003, pp. 61–70.

4. M. D. Galanis, G. Theodoridis, S. Tragoudas, and C. E. Goutis, "A high-performance data path for synthesizing DSP kernels," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 25, no. 6, pp. 1154–1162, Jun. 2006.

5. K. Compton and S. Hauck, "Automatic design of reconfigurable domain- specific flexible cores," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 16, no. 5, pp. 493–503, May 2008.

6. S. Xydis, G. Economakos, and K. Pekmestzi, "Designing coarse-grain reconfigurable architectures by inlining flexibility into custom arithmetic data-paths," Integr., VLSI J., vol. 42, no. 4, pp. 486–503, Sep. 2009.

7. S. Xydis, G. Economakos, D. Soudris, and K. Pekmestzi, "High perfor- mance and area efficient flexible DSP datapath synthesis," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 3, pp. 429–442, Mar. 2011.

8. G. Ansaloni, P. Bonzini, and L. Pozzi, "EGRA: A coarse grained reconfigurable architectural template," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 6, pp. 1062–1074, Jun. 2011.

9. M. Stojilovic, D. Novo, L. Saranovac, P. Brisk, and P. Ienne, "Selective flexibility: Creating domain-specific reconfigurable arrays," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 5, pp. 681–694, May 2013.

10. R. Kastner, A. Kaplan, S. O. Memik, and E. Bozorgzadeh, "Instruction generation for hybrid

reconfigurable systems," ACM Trans. Design Autom. Electron. Syst., vol. 7, no. 4, pp. 605–627, Oct. 2002.

11. T. Kim and J. Um, "A practical approach to the synthesis of arithmetic circuits using carry-save-adders," IEEE Trans. Comput.- Aided Design Integr. Circuits Syst., vol. 19, no. 5, pp. 615–624, May 2000.

12. Hosangadi, F. Fallah, and R. Kastner, "Optimizing high speed arithmetic circuits using three-term extraction," in Proc. Design, Autom. Test Eur. (DATE), vol. 1. Mar. 2006, pp. 1–6.

13. K. Verma, P. Brisk, and P. Ienne, "Data-flow transformations to maximize the use of carry-save representation in arithmetic circuits," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 10, pp. 1761–1774, Oct. 2008.

14. S. Xydis, G. Palermo, and C. Silvano, "Thermal-aware datapath merging for coarse-grained reconfigurable processors," in Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE), Mar. 2013, pp. 1649–1654.

15. M. Srinivasaperumal, G. Naveen Balaji, M. Jagadesh "Heterogenous Node Recovery from crash in wireless Sensor actor networks" International Journal of Modern Trends in Engineering and Science,Vol. 3, Issue 6 (2016) pp: 116-120, ISSN: 2348-3121

16. G.Naveenbalaji, N.V.Harisuriya, S.Anandvikash, B.Adithya, S.Arunkumar "Cost effective power supply based on transformer-less circuitry using bridge rectifier" International Journal of Engineering Research,Vol. 4, Issue 3 (May June 2016) pp: 70-74, ISSN: 2321-7758

17. G. Naveen Balaji, S. Chenthur Pandian, D. Rajesh "A survey on effective Automatic Test Pattern Generator for self-checking Scan - BIST VLSI circuits" International Research Journal of Engineering and Technology,Vol. 3, Issue 5 (May 2016) pp: 645-648, ISSN: 2395 -0056

18. G.Naveen Balaji, V.Aathira, K. Ambhikavathi, S. Geethiga, R. Havin "Combinational Circuits Using Transmission Gate Logic for Power Optimization" International Research Journal of Engineering and Technology,Vol. 3, Issue 5 (May 2016) pp: 649-654, ISSN: 2395 -0056

19. R. ArunSekar, G. Naveen Balaji, A. Gautami, B. Sivasankari "High Efficient Carry Skip Adder in various Multiplier Structures" Advances in Natural and Applied Sciences (Annexure II), Vol. 10 Issue 14 (Special) (Oct 2016) pp: 193-197, ISSN: 1995-0772

20. M. Srinivasaperumal, K. Boopathi Raja, G. Naveen Balaji, E. Christina Dally "Concurrent Node Recovery From Failure In Wireless Sensor-Actor Networks" KARI Research Journal, Vol. 1 Issue 4 (Oct - Dec 2016) pp: 28-33, ISSN: 2456-6136

21. G. Naveen Balaji, V. Narayanan,V.S. Nivash "Low Power and High performance JK Flip – Flop using 45 nm Technology" International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE) Vol 3, Issue 10, October 2016, pp:26-29, ISSN: 2394-6849

22. M. Srinivasaperumal, K. Boopathi Raja, G. Naveen Balaji, E. Christina Dally "Concurrent Node Recovery From Failure In Wireless Sensor-Actor Networks" Advances in Natural and Applied Sciences (Annexure II), Vol. 10 Issue 17 (Dec 2016) pp: 240-246, ISSN: 1995-0772

23. G. Naveen Balaji, R. Prabha, E. Shanthini, J. Jayageetha, Mohand Lagha "Rapid low power Synchronous circuits using transmission gates" Advances in Natural and Applied Sciences (Annexure II), Vol. 10, Issue 17 (Dec 2016) pp: 287-291, ISSN: 1995-0772

24. G. Naveen Balaji, S. Chenthur Pandian, D. Rajesh "Fast Test Pattern Generator using ATALANTA M 2.0" Asian Journal of Research in Social Sciences and Humanities (Annexure I) Vol. 7 No. 2 (Feb 2017) pp. 721-729 ISSN: 2249-7315 DOI: 10.5958/2249-7315.2017.00124.1

25. G. Naveen Balaji, V. Aathira, K. Ambhikavathi, S. Geethiga, R. Havin "Low Power and High Speed Synchronous Circuits using Transmission Gates" Asian Journal of Research in Social Sciences and Humanities (Annexure I), Vol. 7 No. 2 (Feb 2017) pp. 713-720. ISSN: 2249-7315, DOI: 10.5958/2249-7315.2017.00123.X

26. G. Naveen Balaji, S. Anusha, J. Ashwini "GPS Based Smart Navigation for Visually Impaired Using Bluetooth 3.0" Imperial Journal of Interdisciplinary Research (IJIR) Vol. 3, No. 3, 2017, pp. 773-776. ISSN: 2454-1362

27. G. Naveen Balaji, N.V. Hari Suriya, S. AnandVikash, R.Arun, S. Arun Kumar "Analysis of Various Liquid Components under Different Temperature and Density Constraints Pertaining To Fractional Distillation" Imperial Journal of Interdisciplinary Research (IJIR) Vol. 3, No. 6, 2017, pp. 664-669. ISSN: 2454-1362

28. G. Naveen Balaji, D. Rajesh "Python Based Reverse Timing Algorithm for Human Brain Activity Using Color Psychology" International Journal of Indian Psychology, Vol. 4, No. 3, DIP: 18.01.111/20170403, pp: 79-86, ISSN 2348-5396

29. G. Naveen Balaji, S. Chenthur Pandian, D. Rajesh "High Performance Triplex Adder Using CNTFET" International Journal of Trend in Scientific Research and Development, Vol.1, No. 5, pp: 368-373, ISSN 2456 - 6470

30. G. Naveen Balaji, S. Chenthur Pandian, S. Giridharan, S. Shobana, J. Gayathri "Dynamic and Non-Linear Charge Transfer Through Opto-Deportation by Photovoltaic Cell" International Journal of Trend in Scientific Research and Development, Vol. 1, No. 5, pp: 486-492, ISSN 2456 - 6470

31. G. Naveen Balaji, S. Karthikeyan, M. Merlin Asha " 0.18µm CMOS Comparator for High-Speed Applications" International Journal of Trend in Scientific Research and Development, Vol. 1, No. 5, pp: 671-674, ISSN 2456 - 6470

32. G. Naveen Balaji, K. Saravanan, R. Poorani, T. Vishnu Priya, R. Reka Raj "Advanced Security System using PIC through Bluetooth" International Journal of Trend in Scientific Research and Development, Vol. 1, No. 5, pp: 675-685, ISSN 2456 - 6470

33. G. Naveen Balaji, N.V. Hari Suriya, S. Anand Vikash, S. Arun Kumar, R. Arun, "Gasoline Purity Meter Using Peripheral Interface Controller for Automobile Applications" International Journal of Engineering and Technical Research (IJETR), Vol. 7, No.10, pp:46-55, ISSN: 2321-0869

34. G. Naveen Balaji, S.Chenthur Pandian, "Design for Testability of Kipbond Logic" "Perspectivas em Ciência da Informação" (Annexure - I), School of Information Science of the Federal University of Minas Gerais (UFMG), Vol. 22, No. SP.01, pp: 261-284, ISSN 1413-9936