ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

DESIGN AND IMPLEMENTATION OF PCM DECOMMUTATOR ON A SINGLE FPGA

G.PRASAD[1] , Dr. N.VASANTHA[2]

Scientist "SF", National Remote Sensing Centre, ISRO, Hyderabad, India.

Professor & Head, Department of Information Technology, Vasavi College of Engineering, Hyderabad, Andhra Pradesh, India

Abstract: After frame synchronization, individual measurands are identified according to the frame location. The decommutator identifies and extracts embedded asynchronous data stream (EADS) words. Thus PCM Decommutator is a very crucial subsystem in the satellite data acquisition unit of satellite ground station. A PCM decommutator is designed and implemented on a Stratix FPGA. The Hardware design comprises of four modules. a) Decommutator b) FPGA on chip memory bank c) Data storage on external FIFO banks d) PCI-X Master IP core integrated on the same FPGA. Decommutator will identify and separate the individual parameters from the incoming satellite PCM stream. M4k memory of the Stratix is used to develop a FPGA on chip Memory module to temporarily store small volume of the decommutated data before sending it to a large FIFO on the board. Altera 64bit Master IP core is integrated into the same FPGA to interface the stored data to a higher end server. A software program is written in Visual C++ to read the data from FIFO and store in the server RAID. The validation of the modules is done with an inbuilt data simulator.

Key words: **Frame Synchronization, Decommutation, Onchip RAM, PCI Master Core**

I. INTRODUCTION

The hardware design consists of four major modules 1) Decommutator 2) FPGA On chip memory bank 3) External FIFO bank 4) PCI-X interface to host server. This design is realized in AHDL and VHDL and the software used is Altera's Quartus. The decommutator parameters are selectable for catering to different satellites and accordingly the on chip memory is configured as 1k x 64 Qwords and data is temporarily written in the memory. Data is read out from the on chip memory and written in a large external FIFO (128k x 72bits)for storing large volumes of data of the order of 2Mbytes. The FIFO used is IDT 72T72115c.The IP core used is Altera MT64 and is used to interface with the PCI-X 64bit bus in DMA mode. The FPGA used to implement the design is Stratix **EP1S25F1020C5** which has a capacity of 25K logic elements, 6 PLLs, 10 DSP blocks, 1,944,576 Memory elements and 706 I/Os. A software program written in VC++ is used to read the data from the PCI IP core and write to the RAID of the server. The selection parameters are written into the IP core and to FPGA for selecting the desired satellite. Simulations and synthesis is done by the Quartus software tool provided by Altera. After successful total compilation the program output file is loaded into the FPGA using a JTAG connector.

II. QUARTUS SOFTWARE AND ALTERA HARDWARE DESCRIPTIVE LANGUAGE

The Altera Quartus II design software is a multiplatform design environment that easily adapts to specific needs in all phases of FPGA and CPLD design. Quartus II software delivers the highest productivity and performance for Altera FPGAs, CPLDs, and Hard Copy ASICs. Quartus II software delivers superior synthesis and placement and routing, resulting in compilation time advantages. Compilation time reduction features include, Multiprocessor support, Rapid Recompile, Incremental compilation. Quartus II Analysis and Synthesis, together with the Quartus II Fitter, incrementally compiles only the parts of your design that change between compilations. By compiling only changed partitions, incremental compilation reduces compilation time by up to 70 percent. For small engineering change orders (ECOs), the Rapid Recompile feature maximizes your productivity by reducing your compilation time by 65 percent on average, and improves design timing preservation.

AHDL is a proprietary digital Hardware Description Language (HDL) from Altera Corporation for programming their Complex Programmable Logic Devices (CPLD) and Field Programmable Gate Arrays (FPGA). This language has an Ada programming language-like syntax and similar operation to VHDL or Verilog. It is supported by Altera's Quartus and Max+ series of compilers. An advantage of AHDL is that all language constructs are synthesizable. AHDL is to Verilog

ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

much as assembly language is to a higher-level programming language: in AHDL, you have more control.

III. FPGA STRATIX EP1S25F1020C5

The Stratix FPGA is used to implement the four modules i.e. Decommutator, on chip memory, external fifo interface & control logic and PCI IP Master core. Stratix devices contain a two-dimensional row- and column-based architecture to implement custom logic. A series of column and row interconnects of varying length and speed provide signal interconnects between logic array blocks (LABs), memory block structures, and DSP blocks. The logic array consists of LABs, with 10 logic elements (LEs) in each LAB. An LE is a small unit of logic providing efficient implementation of user logic functions. LABs are grouped into rows and columns across the device.

M512 RAM blocks are simple dual-port memory blocks with 512 bits plus parity (576 bits). These blocks provide dedicated simple dual-port or single-port memory up to 18-bits wide at up to 318 MHz. M512 blocks are grouped into columns across the device in between certain LABs. M4K RAM blocks are true dual-port memory blocks with 4K bits plus parity (4,608 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 36-bits wide at up to 291 MHz. These blocks are grouped into columns across the device in between certain LABs. M-RAM blocks are true dual-port memory blocks with 512K bits plus parity (589,824 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 144-bits wide at up to 269 MHz. Several M-RAM blocks are located individually or in pairs within the device's logic array.

Digital signal processing (DSP) blocks can implement up to either eight full-precision 9 × 9-bit multipliers, four full-precision 18 × 18-bit multipliers, or one full-precision 36 × 36-bit multiplier with add or subtract features. These blocks also contain 18-bit input shift registers for digital signal processing applications, including FIR and infinite impulse response (IIR) filters. DSP blocks are grouped into two columns in each device. Each Stratix device I/O pin is fed by an I/O element (IOE) located at the end of LAB rows and columns around the periphery of the device. I/O pins support numerous single-ended and differential I/O standards. Each IOE contains a bidirectional I/O buffer and six registers for registering input, output, and output-enable signals. When used with dedicated clocks, these registers provide exceptional performance and interface support with external memory devices such as DDR SDRAM, FCRAM, ZBT, and QDR SRAM devices. High-speed serial interface channels support transfers at up to 840 Mbps using LVDS, LVPECL, 3.3-V PCML, or HyperTransport technology I/O standards.

III. IP CORE PCI_MT64 MEGACORE FUNCTION

The IP core provides an interface between the Altera pci_mt64 MegaCore function and a 64-bit, 2-MByte FIFO module. It Supports 32- and 64-bit PCI master and target transactions, Supports chaining and non-chaining mode DMA, Uses the dual-port FIFO buffer function from the library of parameterized modules (LPM), This design shows how to connect the local-side signals of the Altera pci_mt64 MegaCore function to local-side applications when the MegaCore function is used as a master or target on the PCI bus. The design consists of the following elements Master control logic, DMA engine, Data path FIFO buffer functions and FIFO interface as shown in Fig 1.

A. Master Control Logic

When the pci_mt64 function is acts as a master, the master control logic interacts with the DMA engine to control the PCI master transactions. During a PCI master write, the data flows from the local master to the PCI bus. The master control logic Provides status of the PCI bus to the DMA engine, Interacts with the pci_mt64 function to execute a PCI master write cycle, Transfers the data from the external FIFO-to-PCI FIFO buffer to the pci_mt64 function.

B. DMA Engine

The DMA engine interfaces with the master control logic, the data path FIFO buffer s, and the FIFO interface to coordinate DMA transfers to and from the FIFO. The DMA engine consists of DMA control logic, DMA registers, DMA descriptor FIFO buffers.

C. DMA Control Logic

The DMA control logic Provides control signals to the master control logic to prompt it to request the PCI bus when needed, Triggers a new access to the external FIFO, Monitors the data path FIFO buffer's and the current FIFO access, Monitors the DMA registers in order to initiate a new transaction, Loads the address counter register (ACR) and byte counter register, (BCR) in the DMA registers when DMA is in chaining mode, Updates the interrupt status register (ISR) and control and status register (CSR) in the DMA registers (chaining and non-chaining mode).

D. DMA Registers

Setting up the DMA registers in the DMA engine initiates DMA transactions. These registers are memory-mapped to BAR0 of the pci_mt64 function; they can be accessed with a target transaction to their memory-mapped addresses. The registers must be written by another master on the PCI bus. The DMA registers consists of Control and status register (CSR), Address counter register (ACR), Byte counter register (BCR), Interrupt status register (ISR), Local address counter.

E. DMA Descriptor FIFO Buffer

The DMA descriptor FIFO buffer provides the storage area for the series of byte count and PCI address pairs when the DMA is programmed to operate in chaining mode. The size of the descriptor FIFO buffer is 256 x 32, and it is capable of holding up to 128 DMA transactions in a chain. This FIFO buffer must

ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

be written with byte count and address pairs by another master/host on the PCI bus before starting the DMA in chaining mode. The descriptor FIFO buffer is read by the DMA control logic to fetch the current byte count to the BCR and address to the ACR before executing the next DMA transaction in a chain.

F. Data Path FIFO Buffers

The data path FIFO buffers serve as the buffer space for the data flowing between the external FIFO and PCI bus. The FIFO buffers are needed to resolve the external FIFO's high data-access latency. The design implements the following FIFO buffer's PCI-to-internal FIFO buffer (128 x 64), Data mask FIFO buffer (128 x 8).

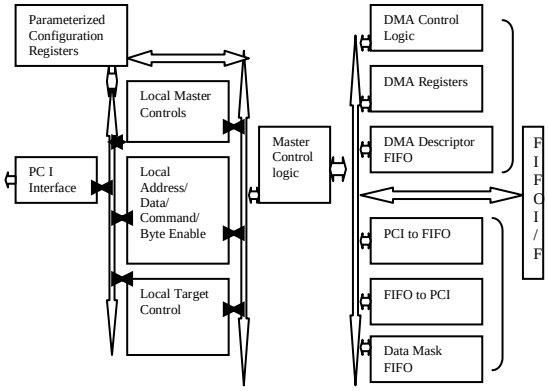

FIG 1. BLOCK DIAGRAM OF PCI –MT64 IP CORE

V. LOGIC IMPLEMENTED IN THE FPGA

The total design comprises of Data Simulator logic, Frame Synchronization logic, Decommutation logic, Control logic, FPGA on chip memory using M4K memory block, external FIFO banks and PCI DMA Core and interface to host server are realized in the 90nm FPGA as shown in Fig 2.

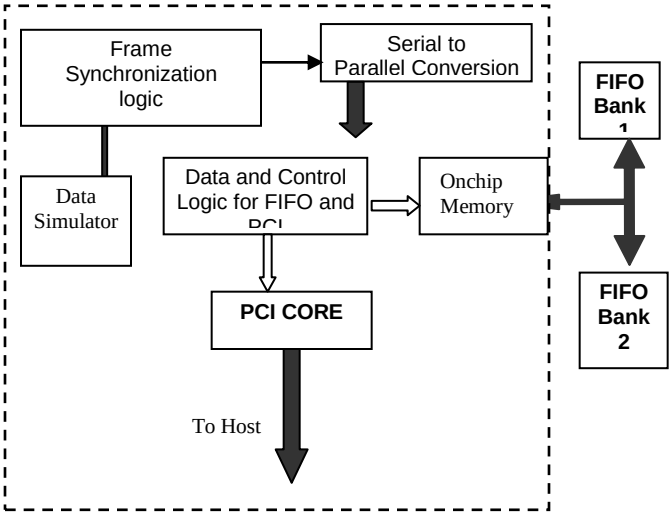

FIG 2. PCM DECOMMUTATOR BLOCK DIAGRAM

The dotted box indicates the modules implemented in the FPGA.

A. DATA SIMULATOR

The Data Simulator is to simulate the data. Data Simulator logic generates the FS code, variable line count in the aux field and fixed video data pattern. A Crystal Oscillator of 105MHz is the source which is divided to generate the required frequency of 52.5MHz. The serial data and clock are connected to a RJ45 connector.

B. FRAME SYNCHRONIZATION LOGIC

The Frame Synchronization logic is realized in the FPGA (Stratix EP1S25F1020C5). The 128 bit correlation function is realized in the FPGA. The incoming data is compared with the reference frame sync code. When the correlation score is >= the Threshold a Frame Sync Detect pulse is generated. To prevent false detects a flywheel logic is included with strategy which has a search, check and lock modes. When two consecutive syncs are detected the logic will change from search to check and later to lock mode. Like wise when a sync loss occurs the logic will change from lock to check and when two consecutive sync loss occur the logic will revert to search mode.

C. SERIAL TO PARALLEL AND BUFFER LOGIC

The serial data after frame sync detection is converted into 64bit (Qword) parallel data. The buffer logic consists of two 128K X 72bit per channel in depth expansion mode. The

ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

FIFO's used are IDT72T72115L10BB. The 64bit Parallel data is written into FIFO and when the FIFO is Half Full data ready status will be indicated to the PCI logic and read logic will be initiated. Thus the read will be taking place continuously since the data is present in the FIFO always.

D. ONCHIP RAM OF FPGA

Simulated data and clock are inputted to the frame synchronizing logic, where the valid frames are detected. After frame synchronization, individual measurands are identified according to the frame location. The decommutator identifies and extracts embedded asynchronous data stream (EADS) words. The serial data is converted to 64bit (Qword) parallel data. In our design the on chip memory was designed using the M 4K RAM block of the Stratix device. The FPGA memory was designed to store 1KQWords (Qword =64bit) and as two banks. Using the alternate buffer concept as shown in Fig 3 the read and write operations were designed so that 1000 words i.e. 3 frames (each frame is 302 Qwords) of data can be written in one buffer and then the write operation will switch to the next buffer and remaining 1000 Q Words will be written. When the first bank writing is complete a control signal will be issued by the memory controller so that the first buffer data can be read on to the external FIFO, thus the 3 frames written in the first bank will be written to the external FIFO by enabling the write control signal and write clock of the external FIFO. After read out from the first on chip memory bank now this bank will be ready to take in the next 3 frames. Next the second memory bank will be writing to the external FIFO and this memory will also be available for next 3 frames. This will result in continuous write and read between on chip memory and external FIFOs. Thus a large memory of desired volume can be realized. Read operation of the external FIFO will empty the FIFO thus in this way required volume of data from multiple inputs can be written and stored.

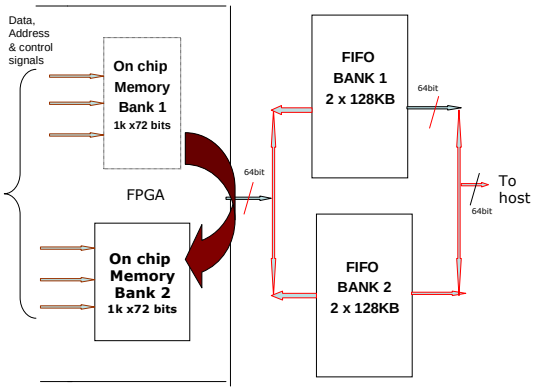

FIG 3. MEMORY IMPLEMENTATION ON FPGA AND INTERFACE TO EXTERNAL FIFO.

Result: Total Memory bits that were required to realize the above design was 573456bits out of 1944576 bits (i.e. 29.49% of the memory bits) and since toggling between high frequency clock (read) and low frequency clock (write) a power saving of 10% was observed.

E. PCI INTERFACE LOGIC.

The PCI interface logic which is working in Master mode to support DMA is incorporated in the FPGA. Parallel data from the FIFO's are read out to the Host Server through this PCI Interface logic operating at 66MHz. The 66MHz clock is derived from the Host server. The 64 bit data read out from the two channels will be written into two files. The Embedded Hardware is integrated with Higher end server with Linux as Operating System.

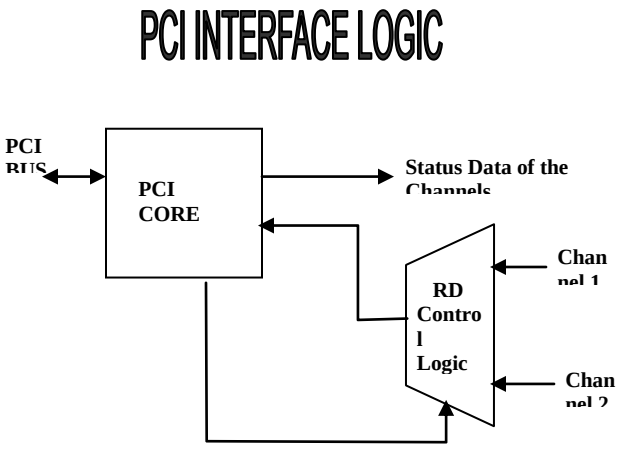

FIG 4. PCI INTERFACING DIAGRAM.

The read control logic will start one channel read out. and place it on the PCI bus as input to the core as shown in Fig 4.. After transferring the required number of bytes an acknowledgement signal from the DMA engine to the read control logic will switch to the second channel and data will be sent as input to the PCI core, this process will continue till the required amount of data is read out.

F. DMA Engine

As shown in Fig 5.
Idle: The channel is idle when no DMA is under progress. A transfer can be programmed and writing to DMA command fields starts a new transfer.
Busy*:* DMA Engine is busy while processing data transfer. It remains so until transfer is either complete or stopped.
Complete*:* When all the data has been transferred, transfer is complete and channel becomes idle on the next clock cycle.

ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

Stop/Error: Transfer ends immediately if a master abort termination occurs during a transfer or if an error is detected in the page descriptor chain.

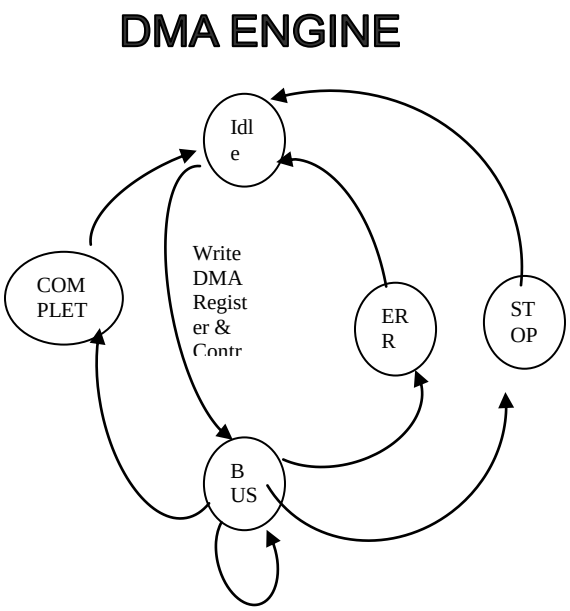

FIG 5. DMA STATE DIAGRAM

Result: 5777 logic elements i.e. 23% of the Stratix FPGA capacity and 318 I/Os i.e. 45% of the FPGA I/O were used to implement the pci interface. Through put of 220Mbytes/ second was achieved. Since the quartus structure file of the core was used directly the routing of all the signals was optimal and hence a power saving of 15% was achieved.

V1. SIMULATION RESULT

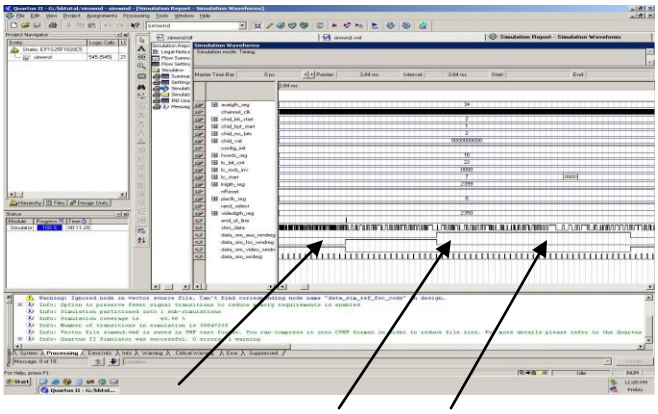

Fig 6. Simulation results of Decommutation logic.

CONCLUSION

The Single FPGA Decommutator, and associated logic designed and developed is suitable for satellite data acquisition systems in the Ground segment. Since the hardware is compact and can be housed in any true 64bit server this forms a embedded hardware and is thus suitable for fixed and mobile applications also. The throughput to the host achieved is between 100 to 200Mbytes/second thus it can cater to high speed data acquisition. Since the major modules are incorporated into the FPGA a power reduction of nearly 20% is achieved in the design. The logic is validated with an inbuilt simulator so that the total chain involved in the design is completely tested.

REFERENCES

[1] J. Toledo, H.Muller, J. Buytaert, F.Bal, A.David, A.Guirao and F.J.Mora (2002), "A plug and play approach to data acquisition", Network architecture and performance digital equipment corporation, Littleton
[2] Bruce A.Wooley, D.Colin Fowlis, James L.Henry and Clarence E.Williams (1979), "An integrated interpolative PCM decoder", IEEE transactions on communications, vol 27.
[3] Charles Geber, Kevin Yee (2000), "Peripheral component interfaces with quick logic QL1624 FPGA", quick logic corporation, Santa Clara.
[4] Michael J Alexander, Gabriel Robens (1996), "New Performance driven FPGA routing algorithms", IEEE transactions on computer aided design vol 15, No23.
[5] Xing Wang, Tapani Ahonen, Jari Nurmi (2000), "A Synthesizable RTL design of asynchronous FIFO", Institute of digital computer systems, Tampere university of technology, Finland.
[6] Sirishsatahaye, K.K Ramakrishnan, Henry Young (1994), "FIFO design for a high speed network interface".
[7] S. Palanivelu, J.Shanmugam Prof and Head, Division of Avionics, Madras Institute of Technology, Chrompet, Chennai "Design and Development of PCM Decommutator with PCI –Interface."
[8] Altera Master Core IP Mt64 User Guide.
[9] DDR and DDR2 SDRAM High-Performance Controllers and ALTMEMPHY IP User Guide section in volume 3 of the External Memory Interface Handbook
[10] On-Chip FIFO Memory Core in Volume 5: Embedded Peripherals of the Quartus II
[11] IDT FIFO Reference Guide
[12] Developing high speed memory interfaces. The Lattice SCM FPGA advantage White paper February 2006.
[13] Memory System Design from Altera Corporation February 2010.
[14] Global Memory Mapping for FPGA based Reconfigurable systems, Iyad Quaiss and Ranga Vemuri.
[15]. Mouzam Khan, Altera Corporation. Power Optimization in FPGA Designs .SNUG San Jose 2006.
[16]. Srinivas Devadas, Massachusetts Institute of Technology, Department of EECS and Sharad Malik, Princeton University Department of EE A Survey of Optimization Techniques Targeting low power VLSI circuits, 32nd ACM/IEEE Design Automation Conference 1995.
[17]. Mouna Nakkar and Paul Franzon. Low Power Logical Element for FPGA Fabric ,2002 IEEE
[18]. Dennis Hudgins, National Semiconductor. Power Supply Design consideration for Modern FPGA's .http://www.eetimes.com/ June 2010.
[19]. Patrick GIRARD, member IEEE Low Power Testing of VLSI circuits: Problems and Solutions

ISSN : 2278 – 1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 7, September 2012

[20]. N.Vasantha, M. Satyam, and K. Subba Rao, "Universal Transitions Count Module for Power Analysis," Journal of Computer Society of India, Vol. 36, No. 2, 2006.
[21]. Differences in logic utilization between Quatus II and Synplify report files.Technical Brief 84, November 2002 ver 1.0
[22]. Improving FPGA Design Speed with Floorplanning *by Consultant Kent Salomon.* Copyright © 2008 Danish Technological Institute
[23]. Stratix Device Handbook, Volume 1 July 2005.

Biography

G.Prasad (M-IEEE, FIETE) received M.Tech degree in Electronics from JNTU Hyderabad in 1995, MBA from IGNOU, New Delhi 2000. He is presently working as Scientist "SF" at National Remote Sensing Centre, ISRO, Hyderabad. His nature of work includes design and development of satellite data acquisition systems, high speed communication between different data acquisition sites through satellite networking. His research interests include VLSI designs, embedded system and realizing systems on programmable chips.

Dr.N.Vasantha (M-IEEE, LM-CSI,IETE,ISTE, M VSI) received the B.E. degree from College of Engineering, Guindy, Madras, in 1977, the M.Tech. degree from JNTU, Hyderabad, AndhraPradesh, in 1986, the Ph.D. degree in Electronics & Communication Engineering from the Osmania University, Hyderabad, AP, in 2008. She is currently working as Professor & Head, Department of Information Technology, Vasavi College of Engineering, Hyderabad, AP. She has taken the initiative to start the value added courses in VLSI Design, Embedded Systems and Digital Signal Processing. Her research interests are in Digital VLSI Design and low-power circuits. A citation and a cash award was given by the Management Vasavi Academy of Education for the same. She is the recipient of the prestigious IETE-Prof. K.Sreenivasan's Memorial Award(2010).