Journal of Computer Science 2 (5) : 388-394, 2006
ISSN 1549-3636

Design of a Merged DSP Microcontroller for Embedded Systems using Discrete Orthogonal Transform

[1]A. K. Rath and [2]P.K.Meher
[1]Department of Computer Science and Engineering
Krupajal Engineering College, Bhubaneswar, Orissa, India
[2]School of Computer Engineering Nanyang Technological University, Nanyang Avenue, Singapore

Abstract: In this paper we have proposed the design of a DSP microcontroller where the processor load is significantly reduced by relegating the math-intensive DSP algorithm to dedicated transform computing modules. It is shown that the use of such transform modules will facilitate scalability, reusability and flexibility for implementation of wide varieties of DSP functionalities. Moreover, it would be possible to meet the need of real-time DSP performance through high throughput computation of orthogonal transforms by pipelining and parallel processing. Use of additional data storage and dedicated buses for DSP functionalities would avoid any possible resource sharing conflicts. The proposed architecture makes only incremental modification to the instruction set of conventional microcontroller. Therefore DSP hardware of the proposed structure may also be used as pluggable core to be used with a microcontroller when DSP algorithms are required to be implemented.

Keywords: Digital Signal Processor, microcontroller, merged architecture, embedded system, core-based DSP

INTRODUCTION

Digital signal processing (DSP) is a word synonymous with the digital revolution that has been taking place in the recent years and has drawn the attention of the community of design engineers in electronic industry. Primarily it deals with the processing of digital data, but it can also be used for manipulation of analog signals after being converted into suitable digital form. DSP-based products have been found successful in several application areas. Now-a-days, it is increasingly used in electronic systems for digital communication and multimedia processing. DSP processors (DSPs) are special purpose devices, designed especially to handle computation-intensive DSP algorithms[1,2]. A DSP processor consists of I/O, data memory, program and control memory, address generators, ALU, multiply-accumulate (MAC) unit/ barrel shifter (Fig. 1). The MAC unit, address generator and barrel shifter are used in DSPs to realize faster implementation of DSP specific computation e.g. digital convolution and filtering. DSPs are widely used for processing of real-time signals including processing of speech and image data. Signal processing functions implemented in the DSP include modulation/demodulation, channel coding/decoding, speech compression, and echo cancellation[3]. In comparison to other types of microprocessors, DSP processors often have an advantage in terms of speed, cost, and energy efficiency. As DSPs become more powerful and less expensive, they are finding their way into more and more of today's electronics products, from cellular phones to mobile CD players and multimedia computers. DSPs are also employed in industrial applications, such as digital process control and motor control systems. The number and variety of products that include some form of digital signal processing has grown dramatically over the last five years. In the recent years DSP processors appear as embedded constituents in consumer, communications, medical, and industrial products[4]. Now the DSPs with

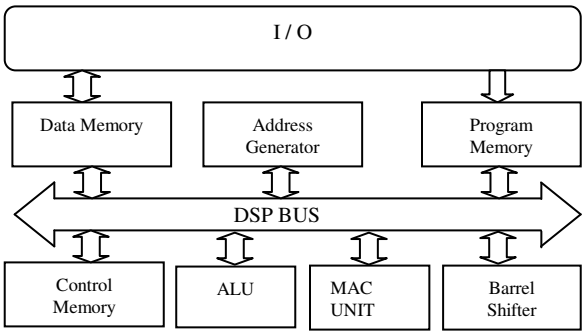

Fig. 1: A typical DSP processor

Corresponding Author : Amiya Kumar Rath, Department of Computer Science and Engineering, Krupajal Engineering College, Bhubaneswar, India.

very high processing power have come down in cost for low end users, and compete with the high end CPUs and micro-controllers.

Several general purpose processors called as microcontroller which are basically designed to execute control-oriented tasks efficiently are available now. These processors are used in control applications where the computational requirements are modest. A microcontroller is a single integrated circuit that contains all the elements of a complete computer system, which includes CPU, memory, input/output ports and other constituent components. The CPU consists of a control unit and an execution unit. It fetches instructions from memory and gets executed in the execution unit. The control unit controls the entire operation by generating appropriate control signals. The execution unit contains set of registers and multiple functional units for performing pipelined execution of instructions. Apart from execution of instructions the microcontroller has also some peripheral functionality. I/O ports are the most basic form of peripheral available on microcontrollers, which behave as interface between the CPU and the external world. The other peripherals, which are included in microcontroller, are A/D and D/A converters to convert real world analog signals in to digital form and vice-versa. As an integrated module it provides address and data bus configuration, system protection, reset and control interrupts. A microcontroller has a wide variety of applications in every imaginable field of industry which ranges from 4-bit micros in low-end applications such as coffee makers to 32-bit RISC micros in automotive engine control. It has got applications in consumer electronics and communication systems, e.g, set-top boxes, cable TV, VCR, digital video disc, camera, stereo, security system, lighting controls etc are used as general consumer products while cellular phone, PDA, pager, FAX/ modem, intercom, cable modem, router, hub etc are used for communication.

DSPs and microcontrollers have several commonalities in their architecture and application domain. Memory organization in both DSPs and embedded controllers are also identical. Like embedded controllers, DSPs are heavily cached and often have on-chip memory controllers to support modern high-performance memory systems. DSPs are also beginning to incorporate peripheral systems that have long been a part of embedded controllers. Many applications require a mixture of control-oriented as well as DSP functionalities. An example of such a system is a digital cellular phone, which must implement both supervisory tasks and voice-processing tasks. In general, microcontrollers provide good performance in controller tasks and poor performance in DSP tasks. DSP processors have the opposite characteristics. Hence, until recently, combination control/signal processing applications were typically implemented using two separate processors: a microcontroller and a DSP processor. A DSP can be used as microcontroller where a microcontroller can be used for executing DSP algorithms. However, using a DSP for simple microcontroller application is not a cost effective choice and a microcontroller in general may not be able to provide the desired real-time math-intensive DSP functionalities.

MERGING OF DSP AND MICROCONTROLLER: EXISTING SCHEMES AND DIFFICULTIES

A lot of architectural differences exist between DSPs and embedded controllers. First, most high performance DSPs contains multiple execution units to achieve the desired speed and performance. The instructions can be executed in the pipeline. Each pipeline has a number of segments, such as load/store, add, multiply, or shift. In contrast to the above, most embedded controllers have only one execution unit. Now-a-days most embedded controllers are RISC (Reduced-instruction-set-computing) processors, while DSPs are moving towards VLIW (Very-long-instruction-word) architectures. A DSP instruction set may be 10 times larger compared to a microcontroller instruction set. In addition, DSPs have many more addressing modes to allow for optimal data handling. Designer always moved to a DSP when the application requires a great deal of mathematical computations since the DSPs perform calculations faster than embedded controller[5].

In the recent years, high performance microcontrollers are available which support DSP functionalities by adding fast multipliers, MAC units, or adding separate DSP units or coprocessors. A number of microcontroller vendors have begun to offer DSP-enhanced versions of their microcontrollers as an alternative to the dual-processor solution. Using a single processor to implement both types of functionalities is attractive, because it can potentially simplify the design task, save total chip area, reduce total power consumption and reduce overall system cost[6,7]. Microcontroller vendors such as Hitachi, ARM (Advanced RISC Machines) and Lexra have taken a number of different approaches to adding DSP functionality to the existing microprocessor design, borrowing and adapting the architectural features common among DSP processors. The DSP units in these microcontrollers contain fast MAC components, barrel shifters, registers, on-chip memory and bit-parallel interfaces to accommodate fast execution of DSP algorithms. However, as the amount of workload increases a single CPU cannot provide the desired performance. So, DSP processors come in to the picture to handle the added load. Embedded microcontrollers can be designed where an existing microcontroller is integrated with the added DSP capability. The loosely

connected combination of microcontroller and DSP was successful, since it performs wide variety of applications. A single merged architecture gives distinct advantages of better and efficient performance and processing power in both application and system development[8]. Many of these hybrid processors achieve signal-processing performance that is comparable to that of low-cost or mid range DSP processors while allowing re-use of software written for the original microcontroller architecture. The fully merged architecture provides simplicity of the single instruction stream and with various forms of parallelism. The merged hybrid architecture with integration of DSPs capability and microcontroller unit, utilizes shared memory and data buses (Fig. 2)[5]. It has however potential threat of access conflict leading to detrimental effect in real-time supervisory and DSP functions.

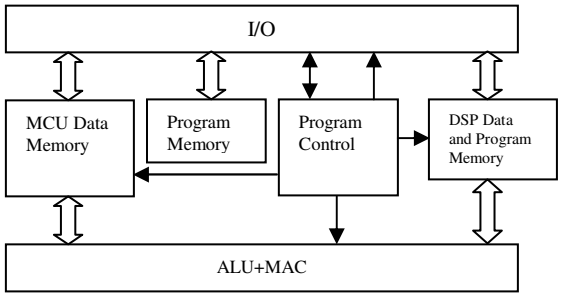

Fig. 2: Fully merged microcontroller and DSP functions in a single processor

The DSP device is built around the digital filter approach where Finite Impulse Response (FIR) filters are used in most DSP applications. FIR filter implementations require a number of address registers to access the stored arrays, typically organized as *X* and *Y* memory and accumulators to sum the results of the multiplications. This approach, therefore, necessitates considerable addition to the hardware complexity for DSP data manipulation with general supervisory control of microcontroller. The increase of control functions of the merged processor may have the risk of failure in real-time applications. Apart from the above, the designer of a DSP microcontroller should take care of cost-to-performance ratio keeping the market segment in view[9]. With the above considerations in the following section we have presented an alternative merged DSP microcontroller using orthogonal transform modules along with fast buffer and arrays of multipliers.

BACKGROUND FOR THE DEVELOPMENT OF THE PROPOSED ARCHITECTURE

Discrete orthogonal transforms like Discrete Fourier transform (DFT), Discrete Hartley transform (DHT) and Discrete Cosine Transform (DCT) can be used for efficient implementation of almost all important DSP functions e.g. FIR filters/convolution, data compression, interpolation, estimation of power spectral density, filter bank realization and wide range of adaptive filtering applications including channel equalization, noise cancellation and system identification[10-14]. In the followings we have shown the use of such DFT and DCT for calculation of convolution/ correlation, implementation of the adaptive filters, interpolation and image data compression.

Computation of convolution using DFT: The computation of finite digital convolution is frequently encountered in several digital signal processing applications. It is used to design and implement finite impulse response (FIR) and infinite impulse response (IIR) filters. Also, it is used to solve difference equations, to compute auto and cross correlations, and to calculate the product of large numbers, as well as, polynomials[15]. The finite digital convolution of two sequences $\{\bar{x}(n)\}$ and $\{\bar{h}(n)\}$ may be computed by the circular convolution of $\{x(n)\}$ and $\{h(n)\}$, which are obtained by appending appropriate number of zeros to $\{\bar{x}(n)\}$ and $\{\bar{h}(n)\}$, respectively, using the overlap-save or the overlap-add technique[16]. The circular convolution of two finite sequences $\{x(n)\}$ and $\{h(n)\}$ of length N is given by

$$y(n) = \sum_{k=0}^{N-1} x(k)h((n-k) \bmod N) \qquad (1)$$

for $n = 0,1, \ldots, N\text{-}1$.

Direct computation of circular convolution according to equation 1 requires N^2 multiplication and $N(N\text{-}1)$ additions. To avoid this computational requirement of brute-force method Fast-Fourier transform (FFT)[17] of Cooley and Turkey is popularly used to compute the circular convolution. By the FFT based method, the operational requirement is reduced to $O(N\log_2 N)$ from $O(N^2)$. The DFT-based approach for computation of circular convolution is depicted in Fig. 3.

The implementation of circular convolution of two N-point sequences will thus involve computation of two numbers of N-point DFT and one N-point Inverse DFT (IDFT), which can be computed by the transform modules of the proposed architecture. The point-by-point multiplications of N-point DFT components can be computed using the multiplier array. Instead of using FFT other discrete orthogonal transform like Discrete Hartley transform may however, be used for implementation of circular convolution[12] of real-valued sequences.

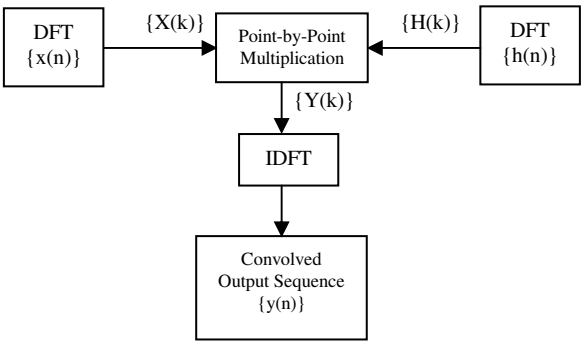

Fig 3: Computation of circular convolution using FFT

Implementation of adaptive filter using DFT: The adaptive digital filter (ADF) is a potential tool for several digital signal-processing applications, such as noise/ echo cancellation, channel equalization and system identification, etc. [18]. The tap delay time (TDL) filter whose weights are updated in every sampling instant according to the Widrow-Hoff least mean square (LMS) algorithm may be considered as the simplest adaptive filter [19]. The block diagram in Fig. 4 depicts working model an adaptive filter, where $x(n)$ is the input, $d(n)$ is the reference output, $y(n)$ is the adaptive filter output. $\{w(n)\}$ constitute the weights of the adaptive filter while $e(n)$ is the error term difference $y(n)$ from $d(n)$.

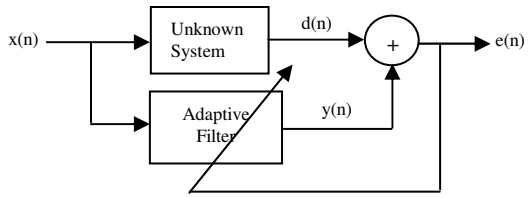

Fig 4: Adaptive filter model

During every iteration the weight values are updated as

$$w_{n+1}(k) = w_n(k) + \mu\, e(n)x(k) \quad (2)$$

for k=0,1……,N-1,

where μ is called as convergence factor,

the error term $e(n) = d(n) - y(n)$ (3)

and the adaptive filter output is give by

$$y(n)=\sum_{k=0}^{N-1} w_n(k)x(n-k) \quad (4)$$

The weight updating by equation (2) and estimation of output $y(n)$ by equation (4) involves most of the computation of the adaptive filter implementation.

The block least mean square (BLMS) adaptive filter proposed by Clark *et. al*[20] is one of the most useful derivatives of the LMS ADF. The conventional LMS ADF accepts a single new input and calculates only one output at a time while the block adaptive filter calculates a finite block of output values from a block of input.

Besides, during every iteration the BLMS ADF updates a block of weights using the block adaptation algorithm. It is shown that[20] the BLMS ADF has convergence properties equivalent to the LMS ADF, but offers considerable saving of computation over the other when the circular convolution and the circular correlation associated with it are performed efficiently.

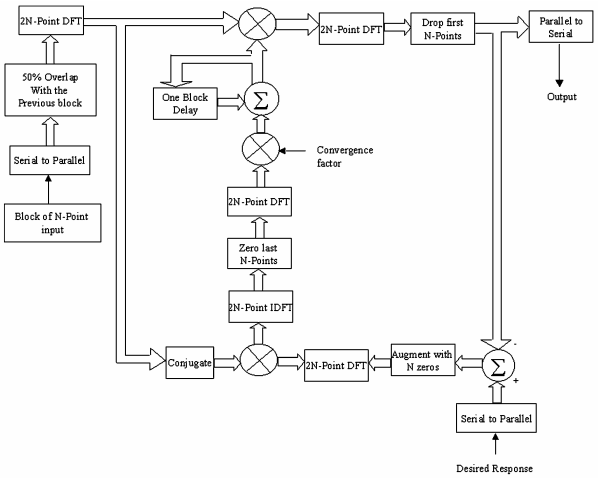

Fig. 5: Transform domain implementation of block LMS

To achieve computational saving, the BLMS ADF has been implemented using the DFT[21]. The implementation of block LMS adaptive filter using discrete Fourier transform is shown in Fig. 5. It involves computation of 3 numbers of 2N-point DFT and 2 numbers of 2N point inverse DFT in every iteration. These DFTs can be implemented in the transform module and the point-by-point multiplication of 2N-point DFT components can be performed in parallel by the multiplier array consisting of 2N multipliers.

Interpolation using DFT: Interpolation, in the context of digital signal processing, is a means to find new samples in between the available samples of a signal, so as to increase the sampling rate according to the requirement[22]. Interpolation can be used for reconstruction of synthetic speech from the signal sampled at a low sampling rate. Therefore, speech data may be sampled at a lower rate for saving the cost of storage and transmission. Interpolation also plays an important role in frequency multiplexing of SSB systems and the digital time-domain beamforming[23]. The theoretical basis for changing the sampling rate by low pass filtering operation is proposed by Schafer and Rabiner[22] and time-domain interpolation using differentiators is introduced by Sudhakar *et. al.*[24]. Efficient schemes using the discrete Fourier transform (DFT) are suggested in the literature for interpolation of discrete-time signals[25-27]. Using the discrete Fourier transform, an interpolated sequence $\{y(n)\}$ of length

K=LN can be obtained from data sequence, $\{x(n)\}$ of length N accordingly following steps.

Step 1: The N-point DFT of input sequence $\{x(n)$, for $n=0,1,2, \ldots N-1\}$ is computed to obtain the DFT components $\{F(k)$, for $k=0,1,2, \ldots, N-1\}$.
Step 2: An intermediate sequence $\{Z(k)\}$, of length $K=LN$, is obtained from the sequence $\{F(k)\}$ by inserting zeros within the sequence $\{F(k)\}$ as given by

$$Z(k) = \begin{cases} L\,F(k) & \text{for } k=0,1,\ldots\ldots N/2-1 \\ L/2\,F(N/2) & \text{for } k=N/2 \\ 0 & \text{for } k=N/2+1\ldots\ldots NL-N/2-1 \\ L/2\,F(N/2) & \text{for } k=NL-N/2 \\ L\,F(k+N-NL) & \text{for } k=NL-N/2+1,\ldots\ldots NL-1 \end{cases}$$

Step 3: K-point IDFT of sequence $\{Z(k)$, for $k=0,1,2, \ldots, NL-1\}$ is computed to obtained the desired interpolated sequence $\{y(n)\}$ of length K.

An interpolated sequence of length $K=LN$ may, therefore, be obtained from data sequence of length N, according to the above steps, while the N-point DFT and K-point IDFT are computed by the transform module of the proposed architecture. The zero padding in the DFT components as required for step 2 can be realized through the interface units.

Image data compression: In the foregoing subsections we have seen that most of computation-intensive DSP functions like convolution, interpolation and adaptive filtering can be realized through DFT. Other orthogonal transforms like the DCT and the DHT can also be used for implementing the above functions. DCT can also be used for some other functions like data compression, sub-band coding and filter bank implementation.

The storage, transmission and processing of uncompressed multimedia content, graphics, audio and video data require considerable storage, bandwidth and computing power for several applications including multimedia communications. Data transfer of uncompressed video data over digital networks requires very high bandwidth even for single point-to-point communication. To achieve feasible and cost-effective solutions, most of the multimedia systems use data compression for digital video and audio data streams. The DCT-based data compression technique is illustrated in Fig. 6.

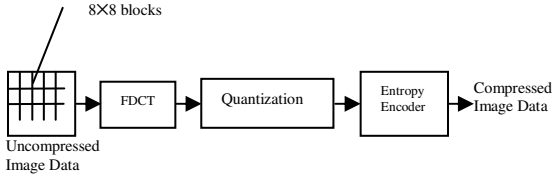

Fig. 6: DCT- based data compression

It involves 4 steps namely block preparation, forward discrete cosine transform (FDCT), Quantization, Entropy encoding. Block preparation includes analog to digital conversion and generating an appropriate digital representation of the information. An image is divided in to blocks of 8 x 8 pixels, and represented by a fixed number of bits per pixel. Picture processing is the second step of the compression process, where a transformation from the time to the frequency domain is performed using DCT. The pixel

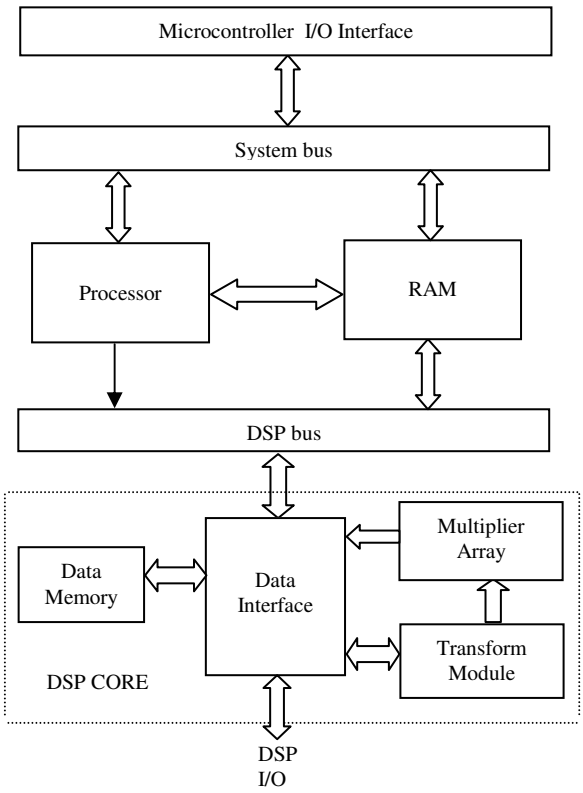

Fig. 7: Proposed DSP microcontroller

values are shifted in to the range –127 to +127 with zero as the center. These data units of 8 x 8 pixel values are defined by $S_{yx,}$ where x and y are in the range of zero to seven. Each of these values is then transformed using 2-D forward DCT (FDCT). On the other end for reconstruction of the original image, the decoder uses the inverse DCT (IDCT) to reproduce the 64 source pixels exactly. In the third step the quantization of all the 64 DCT-coefficients is performed. Entropy encoding is the last step in this process[28]. Hence most of the computational need in the image data compression is centered around the calculation of DCT, which can be taken care of by a DCT processor.

The proposed merged architecture: In the previous Section we have seen that most of the computation intensive DSP functions can be implemented through computation of discrete orthogonal transforms like DFT

and DCT. Moreover, the DSP applications using such transforms also very often involve point-by-point multiplications of DFT components. In the proposed architecture, we have therefore included a transform computation module along with an array of multiplier as shown in Fig. 7. Additional memory for data caching is used in the architecture for buffering the input and output signal values, while the data interface is used for serial to parallel conversions or vice versa as and when required. The proposed structure contains usual auxiliary control components such as system clock generator, timer and serial units as usual for microcontroller functions with processor, primary memory, I/O interface, DMA control and system buses.

The architecture will require inclusion of only a few instructions to the instruction set of a conventional microcontroller because the math-intensive DSP functionalities will be realized by computation of discrete orthogonal transforms such as DFTs, parallel execution of multiplication and parallel data transfer by vector instruction broadcasting. One may perform the computation of the transforms like DFT, DHT and DCT by embedding a fast routine like Fast Fourier Transform (FFT), fast Hartley transform (FHT) or fast cosine transform (FCT), respectively, in the FPGA for low cost implementation. Efficient pipelined schemes for the implementation of these discrete orthogonal transforms in dedicated VLSI chips have been reported in the literature[13-14]. The transform module thus can be realized in a dedicated ASIC or FPGA using such a pipelined architecture to have fast implementation for real-time applications. The transform modules may however be designed to be a single array structure or multi-array structure depending on the speed, cost and size constraints of the embedding environment.

CONCLUSION

Merged architecture to realize digital signal processing and microcontroller functionalities have gained considerable popularity in the past few years in the embedded system arena due to various commonalities in their structure and common presence in several domain of applications. In this paper we have presented a merged DSP microcontroller architecture, where math-intensive functions of algorithms are relegated to a DSP component comprised of a transform modules, a multiplier array storage modules and a data interface unit. The DSP components can be integrated with microcontroller components to form a system-on-a-chip. The use of such transform modules will facilitate scalability, reusability and flexibilities for wide varieties of DSP functionalities. Desired speed performance can be achieved by exploiting the parallelism inherent with the computation of orthogonal transforms in pipelined arrays so as to cater to the need of real-time performance. Additional data storage and dedicated buses for DSP functionalities have been suggested to avoid possible conflict in resource sharing. The proposed architecture makes only incremental modification to the instruction set of conventional microcontroller. Therefore, the DSP hardware of the proposed structure may also be used as pluggable core to be used with a microcontroller when DSP algorithms are required to be implemented[29]. The proposed merged architecture will be simple to design so as to take care of short-time-to market of the evolving embedded products. Apart from that using FPGA based transform modules it can be programmable for flexible custom solutions to domain specific applications[1].

REFERENCES

1. Fettweis, G., 1997. DSP Cores for Mobile Communications: Where are we going? Proc. of ICASSP, pp: 279-282.
2. Verbauwhede, I. *et al.*, 1996 A low-power DSP engine for wireless communications. VLSI Signal Processing IX, IEEE, Eds. W. Burleson *et al.*, pp: 469-478.
3. Lee, E.A., 1988-89 Programmable DSP architectures: Part I & II. IEEE ASSP Magazine.
4. Lapsley, P., J. Bier, A. Shoham and E.A. Lee, 1996. DSP processor Fundamentals: Architectures and Features. IEEE Press.
5. Garreau, O. and R.E. Owen, 1998. Merged architecture approach embeds digital signal processing and improves real-time performance of microcontrollers. Proc. Paper #407 Embedded Systems Conf.
6. Walsh, D., 1996. Piccolo - The ARM architecture for signal processing: An innovative architecture for unified DSP and microcontroller processing. Proc. Intl. Conf. Signal Process. Applications and Technology (ICSPA96), 1 : 658-663.
7. Martin, D. and R. Owen, 1998. A RISC architecture with uncompromised digital signal processing and microcontroller operation. IEEE Intl. Conf. Acoustic Speech and Signal Processing (ICASSP98), pp: 3097-3100, Seattle, WA.
8. Rath, A.K., 2004. Core-based design of embedded DSP system. Ph.D. Thesis. Utkal University, Bhubaneswar.
9. Karthikeyan, M. *et al*., 2000. A framework for cost vs. performance tradeoffs in the design of digital signal processor cores. 13th Intl. Conf. VLSI Design.
10. John, G.P. and D.G. Manolakis. Digital Signal Processing: Principles, Algorithms, and Applications. 3rd Edn.

11. Clark, G.A, S.R. Parker and S.K. Mitra, 1983. A unified approach to time- and frequency- domain realization of FIR adaptive digital filters: IEEE Trans. On Acoustics, Speech, & Signal Processing, ASSP-31 : 1073-1083.
12. Meher, P.K. and G. Panda, 1995. Fast computation of circular convolution of real-valued data using prime factor fast hartley transform algorithm. J. IETE, 41 : 261-264.
13. Nayak, S.S. and P.K. Meher, 1999. High-throughput VLSI implementation of discrete orthogonal transform using bit-level vector-matrix multiplier. IEEE Trans., Circuits and Systems-II: Analog and Digital Signal Processing, 46: 655-658.
14. Maharana, G. and P.K. Meher, 2000. Parallel algorithms and systolic architectures for 1- and 2-D interpolation using discrete transform. Intl. J. Computers & Applications, 22 : 1-7.
15. Agarwal, R.C. and J.W. Cooley, 1997. New algorithms for digital convolution. IEEE Trans. on Acoustics, Speech, and Signal Processing, 25 : 392-410.
16. Oppenheim, A.V. and R.W. Schafer, 1975. Digital Signal Processing. Englewood Cliffs, New Jersey, Prentice-Hall.
17. Cooley, J.W. and J.W. Tukey, 1965. An algorithm for the machine calculation of complex Fourier series. Math. Comp., 19 : 297-301.
18. Cowan, C.F.N. and P.M. Grant (Eds.), 1985. Adaptive Filters: Englewood Cliffs, New Jersey, Prentice-Hall.
19. Widrow, B. and S.D. Stearn, 1985. Adaptive Signal Processing. Englewood Cliffs, New Jersey, Prentice-Hall.
20. Clark, G.A., S.K. Mitra and S.R. Parker, 1981. Block implementation of adaptive digital filters. IEEE Trans. On Circuits and Syst., CAS-28 : 584-592.
21. Clark, G.A., S.R. Parker and S.K. Mitra, 1983. A unified approach to time- and frequency-domain realization of FIR adaptive digital filters. IEEE Trans. on Acoustics, Speech, & Signal Processing, ASSP : 1073-1083.
22. Schafer, R.W. and L.R. Rabiner, 1973. A digital signal processing approach to interpolation. Proc. IEEE, 61: 692-702.
23. Pridham, R.G. and R.A. Mucci, 1979. Digital interpolation beamforming for low-pass and band pass signals. Proc. IEEE, 67.
24. Sudhakar, R., C. Ramesh and S.C. Dutta Ray, 1982. Time domain interpoloation using differentiators: IEEE Trans. Acoust. Speech and Signal Processing, ASSP-30 : 992-997.
25. Yeh, M., J.L. Melsa and D.L. Cohn, 1982. A direct FFT scheme for interpolation decimation, and amplitude modulation. In Proc. 16th Asilomar Conf. Circuits, Syst., Comput., pp: 437-441.
26. Prasad, K.P. and P. Satyanarayana, 1986. Fast interpolation algorithm using FFT. Electron. Lett., 22: 185-187.
27. Adams, J.W., 1987. A subsequence approach to interpolation using the FFT: IEEE Trans. Circuits Syst., CAS-34: 568-570.
28. Richardson, I.E.G., 2002. Video Codec Design. John Wiley and Sons.
29. Rath, A.K. and P.K. Meher, 2001. Embedded DSP microcontroller using discrete orthogonal transform. Proc. XXXVI Ann. Convention, CSI 2001, Kolkata, pp: c-7-c-11.