

## A BIT-SERIAL ARCHITECTURE FOR 1-D MULTIPLIERLESS DCT

*S. Timakul and S. Choomchuay*

Department of Electronics, Faculty of Engineering, and  
 Research Center for Communications and Information Technology (ReCCIT)  
 King Mongkut's Institute of Technology Ladkrabang (KMITL)  
 Chalongkrung Road, Ladkrabang, Bangkok 10520, Thailand  
 Tel: +66 2326 4222 Ext. 114, Fax: +66 2739 2398 Email: kchsomsa@kmitl.ac.th

### ABSTRACT

This paper describes the implementation of a multiplierless 1-D DCT. It is based on the binary DCT where the lifting parameters are approximated with adds and shifts. A bit-serial architecture was investigated in the implementation of such an algorithm. The number of bits requires for specific MSE is also reported.

### KEYWORDS

DCT, Multiplierless DCT, binDCT, VLSI

### 1. INTRODCUTION

The Discrete Cosine Transform (DCT) is significantly of interest in the area of image compression according to its high compaction energy. It has become the core of many international standards such as JPEG, H.26x and the MPEG family [1-3]. In both software and hardware implementations, there appear many fast algorithms to speed up the computation of DCT. A 2-D DCT can be easily computed by recursively used of a 1-D DCT computing scheme. However, the direct implementation of 2-D DCT is generally requires more efforts.

Most DCT computations require floating point multiplications, which indeed slow and clumsy. Such a mathematical notation can be avoided by using Integer implementations, which are usually based on distributed arithmetic [4]. However it stills accompany some drawbacks since these fixed-point multiplications need rather wide data bus (32-bit, for instance). This can lead to a limitation of low power applications such as hand-held devices.

Based on Chen's factorisation of the DCT matrix [5], Tran et al. [6-7] have proposed an approximation computation of DCT by introducing the lifting scheme. The basis multiplication is approximated by the rationals of the form  $k/2^m$ , which can be implemented efficiently by binary shifts. This multiplierless type DCT is also known as a binary DCT or bin DCT. Both the forward and the inverse transforms can be implemented in the similar manner. The implementation can be made further less complicated and more regular by making used of the scaled DCT and in-place computation.

Our work concerns the implementation of a 1-D DCT based on Liang and Tran's work [7]. For very low bit rate applications with quite high compression gain, we treated our design by making used of bit-serial computation scheme. The resulted design is compact and low power consumption.

In section 2, we will outline the fast DCT techniques. A multiplierless approximation binDCT proposed by [7] is greatly reviewed. Our approximation is detailed in section 3. Effects of different word-length computation were also investigated. In section 4, we demonstrated the use of a bit-serial architecture in the implementation of such a binDCT. The simulation results of both software and hardware are given.

### 2. DCT COMPUTATION

#### 2.1 Fast DCT Technique

The one-dimensional (1-D)  $N$ -point DCT, forward and inverse transforms are defined as (1) and (2) below, which are also recommended by [1] for its 8-point DCT.

$$X(k) = \alpha(k) \cdot \sqrt{\frac{2}{N}} \cdot \sum_{n=0}^{N-1} x(n) \cos\left\{\frac{(2n+1)k\pi}{2N}\right\} \quad (1)$$

$$x(n) = \sqrt{\frac{2}{N}} \cdot \sum_{k=0}^{N-1} X(k) \cdot \alpha(k) \cos\left\{\frac{(2n+1)k\pi}{2N}\right\} \quad (2)$$

for  $0 \leq k \leq N-1$  and  $0 \leq n \leq N-1$ . Where

$$\alpha(i) = \begin{cases} \sqrt{\frac{1}{2}} & i = 0 \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

Direct computation of (1) is computation intensive; order of  $N^2$ . Many fast computations were implemented by matrix-matrix multiplication (MMM) or fast algorithm (FCT) or combination of both. Chen et al. [4] proposed a method that vastly reduces the computations; to the order of  $N \log_2 N$ . Because of its fewer operations, not only of the speed that can be increased but also the hardware complexity that can be reduced. The  $N \times N$  matrix calculation can be decomposed into two sets of equations. The first set contains even indexed terms and the second set contains odd indexed terms. When  $N = 2^m$ , for any integer  $m$ , the decomposition can be continued down to 2-point DCT.

The resulted Chen's factorisation of an 8-point DCT is depicted in Fig.1. The computation requires 13 multiplications and 29 additions [7]. The butterfly-like computing scheme and plane rotation enables an in-place computation. It should be noted that the scaling factor of  $1/2$  has to be applied at the end of the transformer to obtain the true DCT coefficients.

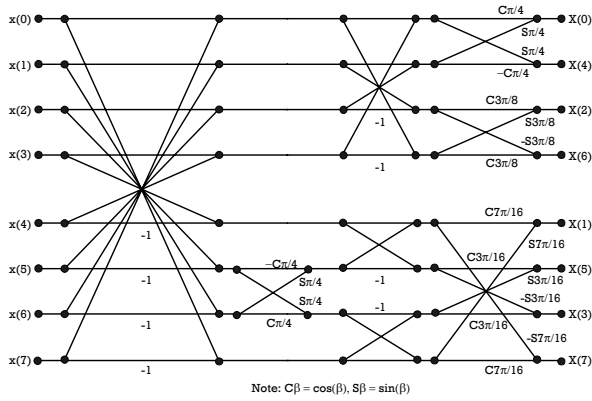


Fig. 1: 1-D Fast DCT Flow Diagram

2.2 Lifting Structures and Binary DCT

A butterfly with rotation plane can be viewed as matrix operation, i.e.  $Y_1 = r_{11}x_1 + r_{12}x_2$  and  $Y_2 = r_{21}x_1 + r_{22}x_2$ . Shown in Fig.2(a), consider the case that  $r_{11} = r_{22} = \cos(\beta)$  and  $r_{12} = -r_{21} = \sin(\beta)$ , with 3-lift lifting structure shown in Fig.2(b), it can be computed that

$$\begin{aligned}
 d &= r_{21} = \sin(\beta) \\
 u_1 &= \frac{r_{22} - 1}{r_{21}} = \frac{\cos(\beta) - 1}{\sin(\beta)} \\
 u_2 &= \frac{r_{11} - 1}{r_{21}} = \frac{\cos(\beta) - 1}{\sin(\beta)}
 \end{aligned}
 \tag{5}$$

The same method can be applied to other butterflies and rotating angles through out the DCT. Each lifting step is a biorthogonal transform, it is hence, reversible. The inverse lifting step is also simple, as what is added in the forward lifting has to be subtracted out in the inverse lifting [7]. With the idea of scale DCT, it is possible to reduce 3-lifting steps to 2-lifting steps with scaling factors. According to the butterfly shown in Fig.2(a), if  $r_{11} \neq 0$  and  $r_{11}r_{22} - r_{21}r_{12} \neq 0$ , then  $u'$ ,  $d'$ ,  $K_1$  and  $K_2$  can be computed from

$$\begin{aligned}
 u' &= \frac{r_{12}}{r_{11}}, & d' &= \frac{r_{11}r_{21}}{r_{11}r_{22} - r_{21}r_{12}} \\
 K_1 &= r_{11}, & K_2 &= \frac{r_{11}r_{22} - r_{21}r_{12}}{r_{11}}
 \end{aligned}
 \tag{6}$$

Shown in Fig.2(c) where  $r_{11} = r_{22} = \cos(\beta)$  and  $r_{12} = -r_{21} = \sin(\beta)$ , then  $u' = \tan(\beta)$ ,  $K_1 = \sin(\beta)$ ,  $d' = -\sin(\beta)\cos(\beta)$  and  $K_2 = 1/\sin(\beta)$ . For some particular rotating angles such as  $\beta \rightarrow j\pi + \pi/2$  for any integer  $j$ , the very small value of  $\cos^2(\beta)$  can have high sensitivity to the round-off and truncation errors. In the same time,  $\tan(\beta)$  becomes bigger and the dynamic range increases. To avoid such an undesirable conditions, the computation in Fig.2(c) can be permuted, as that shown in Fig.2(e) and 2(f).

In summary, if  $\cos^2(\beta) > \sin^2(\beta)$ , the original form (Fig.2(c), 2(d)) is used, and if  $\cos^2(\beta) < \sin^2(\beta)$ , then the permutation form (Fig.2(e), 2(f)) is used instead. Applying lifting scheme to the original DCT shown in Fig.1, the data flow diagram of the forward and reverse lifted DCTs can be drawn as shown in Fig.3 and Fig.4 respectively. Scaling factors  $S_0 \rightarrow S_7$  and  $S'_0 \rightarrow S'_7$  are also the same as those given in [7].

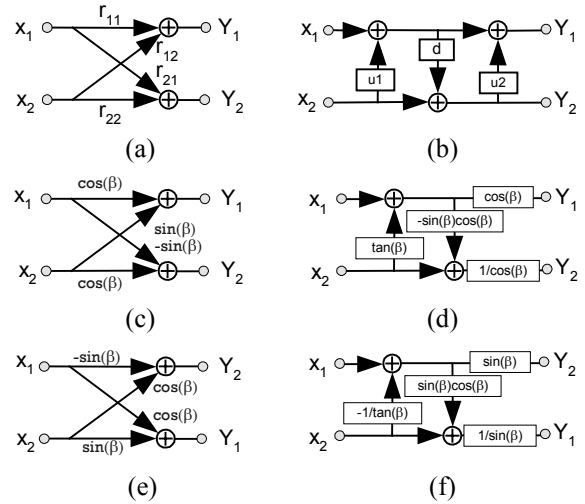


Fig. 2: (a) General butterfly (b) 3-lifting steps structure (c), (d) Lifting steps with scaling (f) Scaled lifting structure for permutation plane rotation (e)

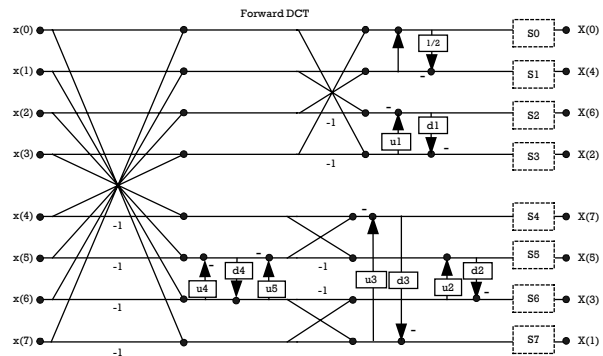


Fig. 3: General Structure of Lifted DCT (Forward Transform) Based on Chen's Factorisation

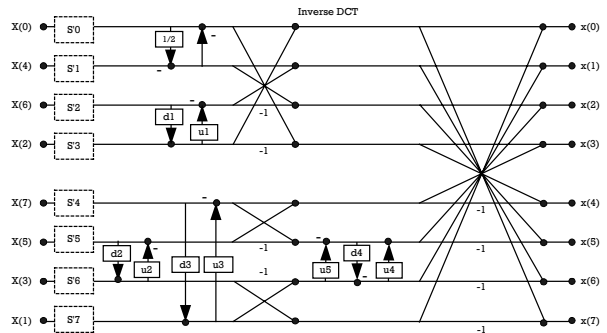


Fig. 4: General Structure of Lifted DCT (Inverse Transform) Based on Chen's Factorisation

According to the above discussion, using the floating-point number format computation stills resource intensive. To reduce such a workload, with resolution trade-off, the fixed-point system is an alternative choice. The binary DCT proposed in [7] is discussed next. The lifting parameters can be approximated by rationals in the format of  $k/2^m$ , where  $k$  and  $m$  are integers. The implementation is then involved binary shifts and adds as it is named binary DCT or binDCT. The scaling values can be omitted. In most case, there is not a major problem of using of this type of transform. However, if the compatibility with the true DCT is needed, the scaling relationship has to be investigated. The complexity of the computation of the binDCT relies upon how close the approximation is made. The bigger number of  $2^m$  yields the better resolutions but the computation requires longer shifts and larger number of bits for storing the intermediate results. We will investigate the effects of number of decimal bits on MSE in the next section. MSE (Mean Square Error) is a widely used in measuring the transformation quality. It is defined as

$$MSE = \frac{1}{(N)^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (F_{ij} - G_{ij})^2 \quad (7)$$

Where  $F_{ij}$  and  $G_{ij}$  are input and output pixel value respectively.

### 3. A BIT-SERIAL MULTIPLIERLESS DCT

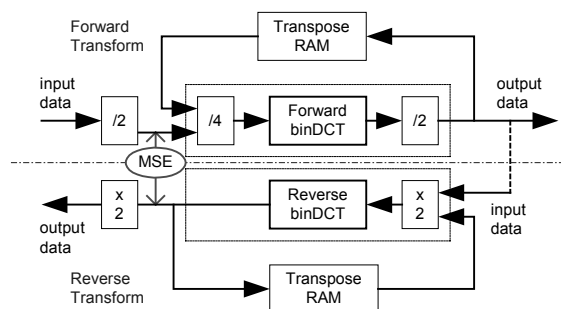
The lifting parameters of the computing topology shown in Fig. 3 and 4 can be approximated to many design sets as given in [7]. Ours are different ones. They are based on the small number of shift and adds as MSE is taken care. To minimise the number of delays and to reduce the signal latency we used the maximum  $2^m = 8$ . The resulted design is given in Table 1 below. Number of shift and adds are counted according to the operations; for instance  $5/8 = 1/2 + 1/8$  is counted as 3 shifts and 1 add since  $1/8$  contains 3 shifts and  $1/2$  contains a single shift. We have 2 approximations, namely, algorithm CA and algorithm CB. Both are limited to maximum of 3 shifts/lift. The CA has more lifts that have been approximated to be closer to their corresponding floating-point values. The CB on the other hand, has less shifts compared to CA. Despite the hardware complexity, both yield the similar performance. With less hardware complexity, the CB holds only 17 shifts and 30 adds in its forward transform scheme.

**Table 1:** Designed Lifting Parameters

Lifters	Values (Function)	Floating-point	CA	CB
u1	$1/\tan(3\pi/8)$	0.414213586	3/8	1/2
d1	$\sin(3\pi/8)\cos(3\pi/8)$	0.353553405	3/8	3/8
u2	$\tan(3\pi/16)$	0.668178623	5/8	5/8
d2	$\sin(3\pi/16)\cos(3\pi/16)$	0.461939762	1/2	1/2
u3	$1/\tan(7\pi/16)$	0.198912392	1/4	1/4
d3	$\sin(7\pi/16)\cos(7\pi/16)$	0.191341738	1/4	1/4
u4	$(-\cos(\pi/4)-1)/\sin(\pi/4)$	0.414213545	3/8	1/2
d4	$\sin(\pi/4)$	0.707106717	3/4	3/4
u5	$(-\cos(\pi/4)-1)/\sin(\pi/4)$	0.414213545	3/8	1/2

It should be noted that the number of shift is defined in quite different manner compared to [7]. Our counts reflects the latency according to delay insertions. Although the fixed-point number representation cannot yield the lossless transformation, we will investigate the number decimal bit that provides the acceptable error for any specific applications. To do this, we examined the number of decimal bits form 0 to 24 bits. This implies the data bus width of 8 to 32 bits. According to our simulation, when the number of decimal bit is greater than 15 the MSE becomes very small (less than  $1e-8$ ). We thus ignore the details in the graph illustrated in Fig.9. We also did compare our approximations to C5 and C6 of [7]. The resulted performance are given in Table 2.

We used 16-bit fixed-point 2's compliment number system (8 bits for decimals) in the implementation of this multiplierless DCT. Input signal is assumed to be grey scale; the pixel value of -128 to +127. We scaled this signal down by 2, before entering the binDCT module. Before any computation, the signal needs further scale down by 4. This is because the 8-point DCT comprises 3 butterfly states and at each state the magnitude may grow twice. We also scaled the results of the final stage by 2 before feeding them into a transpose RAM. This is in fact to avoid the overflow. The arrangement is demonstrated as shown in Fig.5. We did not take into account the scaling values appear at the end of the final lifting states. We will rather modify the quantisation table.



**Fig.5:** Block Diagram of Data Arrangement

### 4. BIT SERIAL ARCHITECTURE

Despite its slow operation, a bit-serial system is fairly simple, low gate counts and require low bandwidth. It therefore suits very well the low power applications where the speed is less crucial.

#### 4.1 A Bit-Serial Adder/Subtractor

A butterfly as well as a lifting scheme computation requires the operations like  $P+Q$  and  $P-Q$ . A bit-serial adder is shown in Fig.6(a). The corresponding LSBs of  $P$  and  $Q$  are summed first, the carry out (if any) is kept in a flip-flop, preparing to add to the sum of next higher weight bits. The flip-flop should be initialised with "0" since there is no carry before LSBs are added. A single adder comprises of a full adder and a delay flip-flop. In term of computation time, a bit-serial adder requires  $n$  clock cycles, where  $n$  is a number of bits in a word.

Subtraction is also as simple as addition, since  $P - Q = P + \overline{Q} + 1$  in the 2's complement number system. An inverter is required to invert  $Q$  and the delay flip-flop is initialised with "1". A bit-serial subtractor is shown in Fig.6(b). It is also easy to combine an adder and a subtractor into a single circuit.

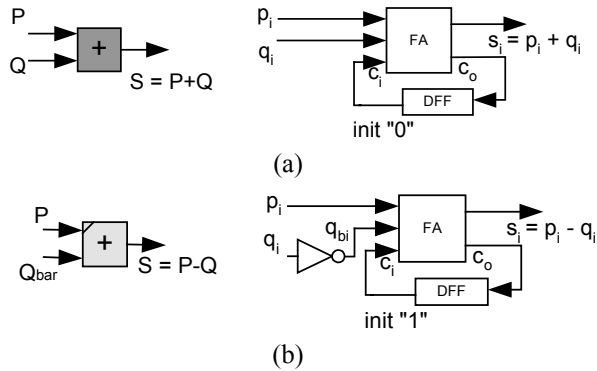


Fig.6: (a) A Bit-Serial Adder (b) A Bit-Serial Subtractor

4.2 Scaling Parameters

The algorithm CB holds the following lifters: 3/8, 5/8, 3/4, 1/4 and 1/2, which can be generalised to  $kP/2^m$ . The computation of  $kP/2^m$  is shown in Fig.7(a) where  $m$  can be 2 or 3. The computation of  $Q \pm kP/2^m$  or vice versa for  $m \geq 3$  is also fairly simple as shown in Fig.7(b). For  $m = 1$  and 2 the arrangement can be made simpler as shown in Fig.7(d).

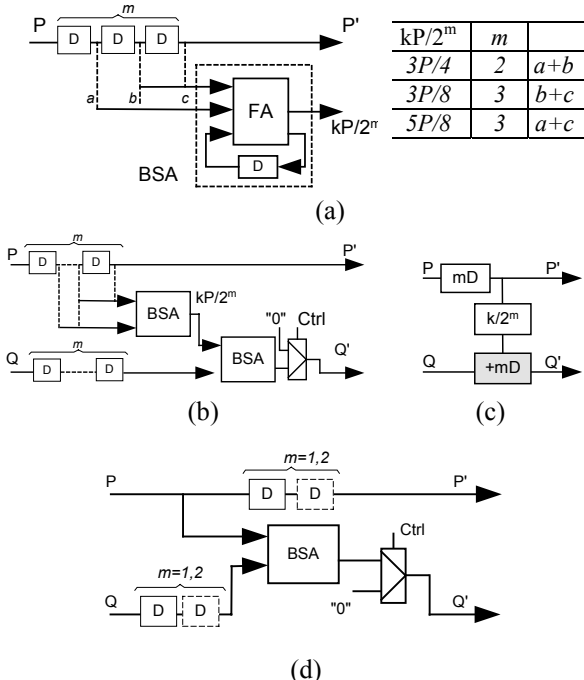


Fig.7: (a)  $kP/2^m$  Scaling Circuit (b) Circuit for Computing  $Q' = Q + kP/2^m$  for  $k=3, 5$  and  $m \geq 2$  (c) Simplified Diagram of (b); (d) Circuit for  $k=1$  and,  $m=1$  and 2

4.3 Bit Slicing Technique

To smooth the data flow, all bit have to be aligned. As the maximum number of shifts is 3, we have assumed three zeros are to be inserted between each data block. The control signal (Ctrl) shown in Fig.7(b) have to discard the sum of the first three bits and replace that with zeroes. The simplified version of Fig.7(b) is shown in Fig.7(c). In additions, we also have to insert delays in path that holds no or holds less lifting operations. The resulted forward transform circuit is shown in Fig. 8.

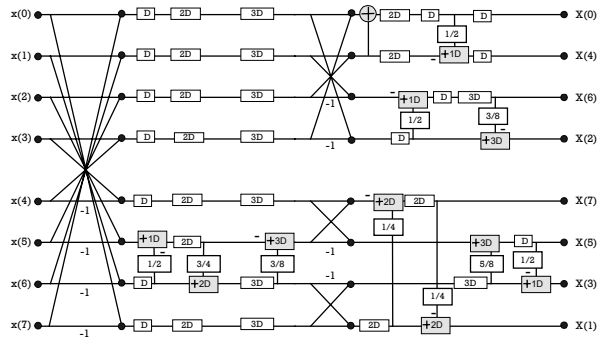


Fig.8: Architecture with Bit Slice Technique

The latency of the above arrangement is 12 clock cycles. Data are flowing in a bit-serial manner, 8 words in parallel. Result is obtained at every clock cycle. As such, the data rate is 8 bits/clock or 8 words/19 clocks. Regardless the data transposition, a 8x8 DCT will take 38 clock cycles. Obviously, the data rate is 64 bits/38 clock cycles. For the image size of 256x256 pixels, the transformation can take 78 ms, if the clock speed of 4 MHz is assumed.

5. RESULTS

MSE of different algorithms are observed via C programming simulation results. The MSE obtained from different decimal bits length are graphically shown in Fig. 9. When the bus-width is greater than 24 bits, the MSE can be made very small and that may claim the near lossless transformation. The performance of CA and CB are comparable to that given by C5 and C6 of [7]. Details are given in Table 2. An obvious advantage we can have; CB is less complexity: - about 60% of C5. Resulted portion of images of different intermediate word length are demonstrated in Fig.10. Architecture shown in Fig. 8 was realized using a bit-serial approach of which the actual implementation can be gate array or full custom design.

Table 2: Performance of BinDCT with Different Approximations (8 bits decimal)

	C5†	C6†	CA	CB
No. of Shifts	30	24	23	17
No. of Adds	36	33	33	30
MSE*	0.001966	0.001837	0.001800	0.001400
MSE**	0.002373	0.002239	0.002137	0.001589

† Based on [7] \* Lena 512x512 pixels \*\* Lena 128x128 pixels

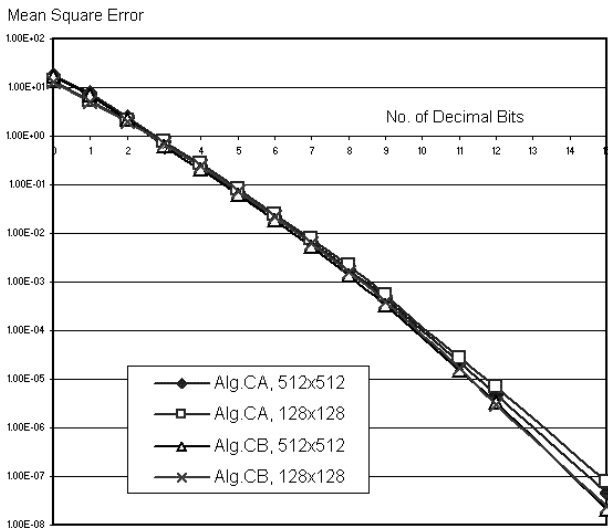


Fig.9: MSE Performance of Algorithms CA and CB

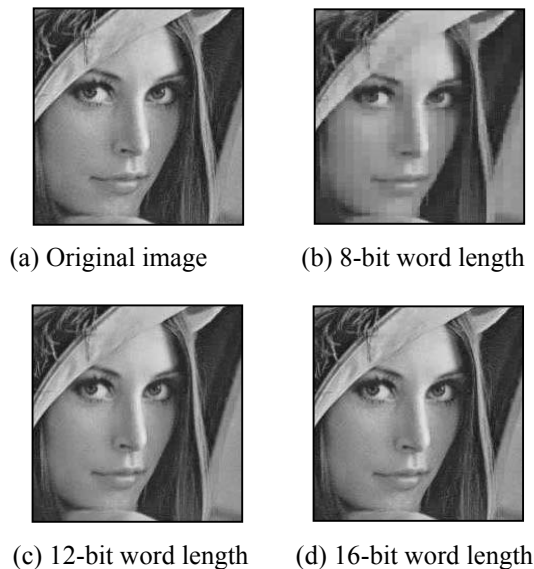


Fig.10: Images at Different Intermediate Word Length (Algorithm CB, Full image size = 512x512 pixels)

## 6. CONCLUSIONS

A binDCT can lead to a simple implementation since multipliers can be avoided and the lifting schemes can be realised by using binary shift and add operations. It tends to be a lossy transformer of which errors are introduced by both lift parameter approximations and finite word length. To reduce the hardware complexity or to increase the computation speed, the lifting parameters have to be simple as  $x/2$ ,  $x/4$  or  $x/8$ . Different sets of approximation can affect the MSE. For a particular arrangement, fine tuning can be made to obtain best result. Lengthening the word length can of course improve both MSE and data rate, but such an effort leads to the increasing of hardware complexity and signal latency.

A bit-serial implementation of the proposed architecture does require minimum hardware. The word length can be easily truncated or extended according to

the precision needed. In our case, a 16 bits fixed-point (with 8 decimal bits) system was simulated and designed. Regardless the complexity of the control circuit which is assumed to be fairly small, only 37 bit-serial adders and about 80 latches are used. With the latency of 12 clock cycles, our design yields the data rate of 1.68-bit/clock cycle. For a fairly low system clock speed, it is possible to meet the specification of low bit rate video, i.e.  $n \times 64$  kbit/sec. For example:  $n = 30$ , the system has to run at the clock speed of  $(30 \times 64 \times 10^3) \times \frac{38}{63} = 1.14$  MHz.

## REFERENCES

- [1] ITU-T recommendation T-81, *Digital Compression of Continuous Tone and Still Images*, September, 1992.
- [2] Coding of moving pictures and associated audio, Committee Draft of Standard ISO 1172: ISO/MPEG 90/176, Dec. 1990.
- [3] Raymond Westwater and Borko Furht, *Real-Time Video Compression; Techniques and Algorithms*, Kluwer Academic Publishers, 1997.
- [4] W. Cham, "Development of Integer Cosine Transforms by the Principle of Dyadic Symmetry," in *Proc. Inst. Elec. Eng., Part I*, Vol. 36, Aug. 1989, pp. 276-282.
- [5] W. H. Chen, C. H. Smith and S. C. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform," *IEEE Trans. Commun.*, Vol. COM-25, pp. 1004-1009, Sept. 1977.
- [6] T. D. Tran, "The BinDCT: Fast Multiplierless Approximation of the DCT," *IEEE Signal Processing Letters*, Vol. 7, No. 6, pp. 141-144, Jun. 2000.
- [7] J. Liang and T. D. Tran, "Fast Multiplierless Approximations of the DCT With the Lifting Scheme," *IEEE Trans. Signal Processing*, Vol. 49, No. 12, pp. 3032-3044, Dec. 2001.

Best Paper Award