Delay efficient carry select logic for N-Point DCT approximation using n/2-point DCT

K. Satya Devi¹, M. Sateesh Babu²

¹VLSI & Embedded systems, ²Dept. of Electronics and Communication Engineering
Kakinada Institute of Engineering and Technology for Women, Korangi, AP, INDIA

Abstract

The Discrete cosine transform (DCT) is prominently utilized as a part of image and video compression. Approximation of discrete cosine transform (DCT) is helpful for diminishing its computational many-sided quality without noteworthy effect on its coding execution. The vast majority of the current algorithms for approximation of the DCT target just the DCT of little transform lengths and some of them are non-orthogonal. This paper shows a summed up recursive algorithm to get orthogonal approximation of DCT where an estimated DCT of length could be gotten from a couple of DCTs of length at the cost of increases for info preprocessing. Proposed algorithm is very versatile for equipment and programming execution of DCT of higher lengths. One exceptionally fascinating element of the proposed plan is that it could be designed for the algorithm of a 32-point DCT or for parallel algorithm of two 16-point DCTs or four 8-point DCTs with a minor control overhead. The proposed design is found to offer numerous preferences as far as equipment multifaceted nature, normality and seclusion and deferral. Test comes about got from FPGA execution demonstrate the upside of the proposed strategy. The many-sided quality is more with equipment when there is increment in number of bits amid move operation. To defeat this issue the design is adjusted with convey select snake. In this paper we proposed the procedure which expands the speed of operation by decreasing the time postponement of operation. This is the upgraded plan approach to decrease the power utilization and many-sided quality.

Keywords: Discrete cosine transform (DCT), Adders, Image Compression, Low complexity Transforms.

I. Introduction

The discrete cosine transform (DCT) is famously utilized as a part of image and video compression. Since the DCT is computationally serious, a few algorithms have been proposed in the writing to register it productively. As of late, huge work has been done to infer rough of 8-point DCT for decreasing the computational multifaceted nature. The primary goal of the approximation algorithms is to dispose of duplications which expend the majority of the power and algorithm time, and to get significant estimation of DCT too. Haweel has proposed the marked DCT (SDCT) for 8 X8 squares where the premise vector components are supplanted by their sign, i.e, 1. Bouguezel-Ahmad-Swamy (BAS) have proposed a progression of techniques. They have given a decent estimation of the DCT by supplanting the premise vector components by 0, 1/2, 1. Moreover, a few image handling applications, for example, following and synchronous compression and encryption require higher DCT sizes. In this specific circumstance, Cintra has presented another class of whole number transforms appropriate to a few square lengths. Cintra have proposed another 16 X16 network likewise for approximation of 16-point DCT, and have approved it tentatively. As of late, two new transforms have been proposed for 8-point DCT approximation Cintra et al. have proposed a low-intricacy 8-point inexact DCT in view of number capacities and Potluri et al. have proposed a novel 8-point DCT approximation that requires just 14 expansion . Then again, Bouguezel have proposed two strategies for augmentation free inexact type of DCT. This transform is a permutated rendition of the WHT which approximates the DCT exceptionally well and keeps up every one of the upsides of the WHT. Proposed two strategies for augmentation free estimated type of DCT. This transform is a permutated variant of the WHT which
approximates the DCT exceptionally well and keeps up every one of the upsides of the WHT.

A plan of approximation of DCT ought to have the accompanying elements:

i) It ought to have low computational unpredictability.

ii) It ought to have low blunder vitality keeping in mind the end goal to give compression execution near the correct DCT, and ideally ought to be orthogonal.

iii) It ought to work for higher lengths of DCT to bolster advanced video coding models, and different applications like following, reconnaissance, and concurrent compression and encryption.

Be that as it may, the current DCT algorithms don’t give the best of all the above three prerequisites. A portion of the current techniques are lacking as far as versatility, speculation for higher sizes, and orthogonality. We mean to keep up orthogonality in the surmised DCT for two reasons. Firstly, if the transform is orthogonal, we can simply locate its opposite, and the portion framework of the backwards transform is acquired by simply transposing the part lattice of the forward transform. This element of opposite transform could be utilized to process the forward and converse DCT by comparative figuring structures.

II. Methodology

ADDPERS

Number juggling is the most seasoned and most rudimentary branch of Mathematics. The name Arithmetic originates from the Greek word ἀριθμός (arithmos). Math is utilized by practically everybody, for assignments going from basic everyday work like checking to cutting edge science and business counts. Accordingly, the requirement for quicker and effective Adders in PCs has been a theme of enthusiasm over decades. Thus, enhancing execution of the computerized snake would significantly propel the execution of double operations inside a circuit bargained of such squares. The execution of an advanced circuit square is gaged by breaking down its energy dispersal, format region and its working velocity.

To people, decimal numbers are anything but difficult to grasp and actualize for performing math. Be that as it may, in computerized frameworks, for example, a chip, DSP (Digital Signal Processor) or ASIC (Application-Specific Integrated Circuit), paired numbers are more down to earth for a given calculation. This happens on the grounds that double values are ideally productive at speaking to many qualities.

Parallel adders are a standout amongst the most basic rationale components inside an advanced framework. Binary adders are one of the most essential logic elements within a digital system. In addition, binary adders are also helpful in units other than Arithmetic Logic Units (ALU), such as multipliers, dividers and memory addressing. Therefore, binary addition is essential that any improvement in binary addition can result in a performance boost for any computing system and, hence, help improve the performance of the entire system.

Fig: 1-bit Half Adder.

The addition operation is represented with + in the above equation. However in Boolean algebra we use the binary notations. For performing different operations we have to use different logic gates like AND, OR, and XOR...etc... In the following documentation the dot operation between two variables represents the ‘AND’ operation i.e ‘a AND b’, similarly, the addition operation between two variables like a + b denotes ‘a OR b’ and a^ b denotes ‘a XOR b’. Considering the situation of adding two bits, the sum s and carry c can be expressed using Boolean operations mentioned above.

\[ S = a \land b \]
\[ C_{i+1} = a \land b \]

\[ 1 \]
The Equation of $C_{i+1}$ can be implemented as shown in Fig.2.1. In the figure, there is a Half adder, with 2 input bits. The longest path from the input to the output is called critical path which is in the solid line. Equation of $C_{i+1}$ can be given to another half adder to perform full adder operation, where there is a carry is taken as input.

$$S_i = a_i \oplus b_i \oplus c_i$$

$$C_{i+1} = a_i \cdot b_i + a_i \cdot c_i + b_i \cdot c_i$$

**RIPPLE CARRY ADDER**

We can be Ripple carry adder by connecting $n$-full adders in cascading. Fig 2.1 shows a 4-bit ripple carry adder. One full adder is responsible for the addition of two binary digits at any stage of the ripple carry. The carry comes from the each full is the input the next full adder. If we want add $n$-bit number then we have to connect the $n$-full adder in cascading.

One of important drawbacks of this adder is that the delay increases linearly with the bit length. Because it propagates the one full adder output carry to another so it takes more time to generate the output, which is approximated by:

$$T = (n-1) t_c + ts$$

(2.14)

**CARRY LOOK-AHEAD ADDER**

Look ahead carry algorithm speed up the operation to perform addition, because in this algorithm carry for the next stages is calculated in advance by using the combination circuit at the every input of the full adder.

**CARRY SAVE ADDER**

The principle reason for convey spare viper to diminishes the expansion of number. The spread postponement depends on the quantity of bits. The convey spare snake comprise of $n$ full adders, every full viper produces total and convey bits for a given data sources.

The whole total is figured by moving the convey succession left by one place and adding a 0 to the front (most noteworthy piece) of the halfway entirety arrangement and including this consequence of grouping with RCA produces the subsequent $n+1$ bit esteem.

This procedure can be proceeded with further, by including a contribution for every phase of full adders, with no moderate convey engendering. These stages can be organized in a twofold tree structure, in the quantity of contributions to be included by utilizing the total postpone logarithmic, and the quantity of bits per information won't be fluctuated. The primary utilization of convey spare calculation is, it is utilized for proficient CMOS usage of much more extensive assortment of calculations and understood for multiplier engineering for fast DSP. Convey spare snake will accelerate the convey engendering in the cluster by
connected in the halfway product offering of exhibit multipliers.

Essentially, convey spare snake is utilized for figuring the total of at least three n-bit paired numbers. Convey spare snake resemble a full viper. Here we are registering aggregate of two 4-bit twofold numbers, so we take 4 full adders at first stage. Convey spare unit comprises of 4 full adders, every full snake comprises of whole and convey in view of the given information. Let X and Y are two 4-bit numbers and creates fractional total and convey as appeared in the beneath:

\[
\text{Fig. 4-bit Carry Save Adder}
\]

**III. Proposed Methodology**

To overcome the complexity here proposed a method in which two carry select adders are utilized. It performs additions twice, one time with the assumption of the carry being “zero” and the other assuming “one”. After calculation of sum, carry of the ripple carry adders is selected by the multiplexer to check whether carry is “0” or “1”. This method reduces the delay of operation, reduces hardware requirement, less complexity and thereby reducing the cost.

**CARRY SELECT ADDER**

A convey select viper is grouped into segments, each of which – with the exception of the minimum critical – performs two increases in parallel, one accepting a convey in of zero, the other a convey in of one. The convey select is planned by utilizing the two four piece swell convey adders. The convey select viper not quick but rather it is straightforward in nature, having an entryway level profundity of \( O(\sqrt{n}) \).

**IV. Software Implementation**

XILINX 12.3
Xilinx software is used by the VERILOG designers for performing Synthesis operation. Any simulated code can be synthesized and configured on FPGA. Synthesis is the transformation of VERILOG code into gate level net list. It is an integral part of current design flows.

**ALGORITHM**

Here the following steps need to be maintained to edit and compile the XILINX

Step 1: Start the ISE Software by double clicking the XILINX ISE icon.

Step 2: Create a New Project by clicking on the new project in the ISE Project Navigator and the following window is generated.

**V. Simulation Results**

**Summary of Carry Select Logic For N-Point Dct Approximation**

The design summary shows the total number of LUTs used in the project. From this we can understand that this project utilizes Delay when compared to the previous design.

![Fig: summary of carry select logic for n-point dct approximation](image1)

**RTL SCHEMATIC OF CSL**

In digital circuit design, register-transfer level (RTL) is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers and the logical operations performed on those signals. RTL is used in the logic design phase of the integrated circuit design cycle. An RTL description is usually converted to a gate-level description of the circuit by a logic synthesis tool. The synthesis results are then used by placement and routing tools to create a physical layout. Logic simulation tools may use a design's RTL description to verify its correctness.

Check Syntax after finally editing the VERILOG source for any errors. After that perform the RTL and TECHNOLOGY schematic for verifying synthesis.

![Fig RTL Schematic of carry select logic for n-point dct approximation](image2)

![Fig Internal RTL Schematic of carry select logic for n-point dct approximation](image3)

**TECHNOLOGY SCHEMATIC**

**OUTPUT WAVE FORMS**

![Fig Technology Schematic](image4)
Fig 7.5: Output Waveforms

RESULTS ANALYSIS

<table>
<thead>
<tr>
<th>LOGIC UTILIZATION</th>
<th>USED</th>
<th>AVAILABLE</th>
<th>UTILIZATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER OF 4 INPUT LUT'S</td>
<td>5329</td>
<td>4656</td>
<td>114%</td>
</tr>
<tr>
<td>NUMBER OF OCCUPIED SLICES</td>
<td>9769</td>
<td>9312</td>
<td>104%</td>
</tr>
<tr>
<td>NUMBER OF BOUNDED IOB’S</td>
<td>2050</td>
<td>232</td>
<td>883%</td>
</tr>
</tbody>
</table>

Table.1 Device Utilization summary of DCT

SYNTHESIS REPORT

Total delay = 49.55
Total memory usage = 344916kb
Number of 4 input LUTs = 5329
Power = 742W

The comparison results with the related work.

<table>
<thead>
<tr>
<th>DESIGN</th>
<th>DELAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXISTING DESIGN</td>
<td>60.170ns</td>
</tr>
<tr>
<td>IMPLEMENTED DESIGN</td>
<td>49.55ns</td>
</tr>
</tbody>
</table>

Table.2 Comparison results with the related work

VI. Conclusion

In this paper, we have proposed carry select adder logic for recursive algorithm to obtain orthogonal approximation of DCT where approximate DCT of length N could be derived from a pair of DCTs of length (N/2) instead of N additions for input preprocessing. The proposed carry select logic for the approximated DCT has several advantages, such as of regularity, structural simplicity, lower-computational complexity, high speed operation and scalability. Along with these we have another advantage that is latency (delay) is reduced by 18.67% then the previous addition techniques. We have also proposed a fully scalable reconfigurable architecture for approximate DCT computation where the computation of 32-point DCT could be configured for parallel computation of two 16-point DCT’s or four 8-point DCT’s.

Future Scope

This can also be extended to N point DCT by using (N/2) point DCT, with reduced latency. We have also proposed a fully scalable reconfigurable architecture for approximate DCT computation where the computation of N-point DCT could be configured for parallel computation of two (N/2)-point DCT’s or four (N/4)-point DCT’s.

References


