VLSI Implementation of Burrows Wheeler Transform for Memory Reduced Distributed Arithmetic Architectures

Remya Ajai A.S., Lintu Rajan, and Shiny C.

Department of ECE, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, Kollam-690525, Kerala, India
{remya.amrita,linturajan08, shyni.c19}@gmail.com

Abstract. Multiply and accumulate function is the important part of digital signal processing algorithms. This can be implemented more effectively with distributed arithmetic (DA) architecture [1]. These architectures make extensive use of look-up tables, which make them ideal for implementing digital signal processing functions on Xilinx FPGAs. An emerging arithmetic-intensive digital signal processing algorithm is the discrete wavelet transform (DWT) which have proven to be extremely useful for image and video coding applications like MPEG-4 and JPEG 2000[2]. But the limitation of this architecture is that the size of look-up tables get increased exponentially as the constant coefficients of wavelet used for these applications increases. In this paper, we proposed a novel methodology to implement the Burrows wheeler transform (BWT) [3] block in FPGA for achieving memory reduced DA.

Keywords: DWT, Burrows Wheeler Transform, Distributed Arithmetic Architecture, Field programmable gate arrays.

1 Introduction

Discrete wavelet transform can be implemented in FPGA using DA architecture [4]. In DA architecture implementation of DWT, the costly multipliers are replaced by shifters and adders. This facilitates reduced power consumption. But the only disadvantage of DA is that the number of look-up table entries increases exponentially with the increase in filter coefficients. Reduced memory DA architectures can be obtained by sorting the table entries using BWT and then compressed using table compression algorithm [5]. At the time of convolution, the required entry can be generated by performing reverse BWT and table decompression [5]. In this paper we implemented the reverse BWT block required for memory reduced DA architecture. The forward BWT can be done offline since the sorted and compressed look-up table is stored in the memory. Hence only the reverse BWT needs to be implemented in FPGA, which will reconstruct the original look-up table entry for a particular inner product operation.

2 BWT Implementation for Memory Reduced DA

Burrows Wheeler Transform (BWT) is a block sorting compression algorithm traditionally used for sorting strings [3]. The transformation is based on the permutation of input sequence. It is called block sorting since blocks are considered for sorting. The look-up table entries for DA arithmetic can be converted into binary and then sorted using BWT so that more number of ones or zeroes may appear continuously. This helps to compress the look-up table using table compression algorithm [5] with a higher compression ratio. BWT is particularly chosen for this application since it is reversible.

3 FPGA Implementation of Reverse BWT

Sorting and compression of look-up table entries are offline processes. Hence only reverse BWT has to be implemented in FPGA. In this section we are discussing the implementation of reverse BWT for generating the required look-up table data for performing the convolution operation from the compressed table.

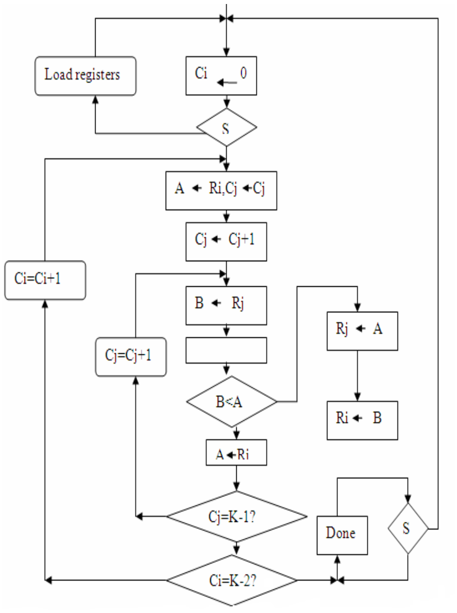

Fig. 1. ASM chart for sorting operation

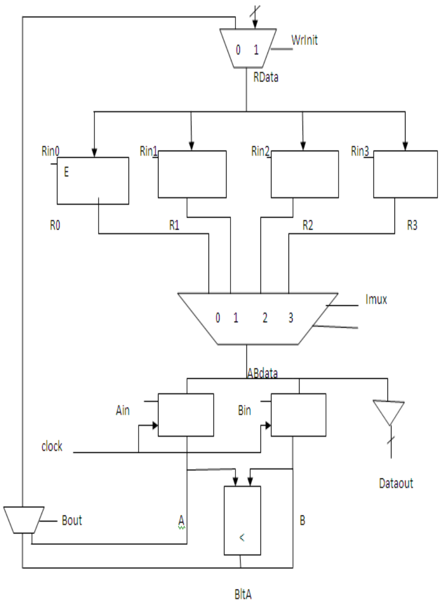

Fig. 2. Data path circuit for sorting

The basic steps include sorting and concatenation. An ASM chart for sorting a list of k unsigned numbers stored in a set of registers R0……Rk-1 is shown in Fig. 1. A data path circuit that meets the requirements of the ASM chart in Fig 1 is illustrated in Fig 2. It shows how the registers R0,…Rk-1 can be connected to registers A and B using 4-to-1 multiplexers. We assume the value k=4 for simplicity. Registers A and B are connected to a comparator circuit and ,through the multiplexers, back to the inputs of the registers R0,….Rk-1.The registers can be loaded with initial data(unsorted)data using the dataIn lines. The data is written (loaded) into each register by asserting the WrInit control signal and placing the address of the register on RAdd input. The tristate buffer driven by the Rd control signal is used to output the contents of registers on Data Out output. The signals Rin0,…….Rink-1 are controlled by the 2-to-4 decoder as shown in figure. If int=1,the decoder is driven by one of the counters ci or cj. If Int=0, then the decoder is driven by the external input RAdd. The signals zi and zj are set to 1if ci=k-2 and cj=k-1.respectively.

4 Implementation and Results

The BWT block implemented is applied for discrete wavelet transform using Daubechies-4 wavelet. The size of the look-up table is reduced from 9216 to 8667 bits.

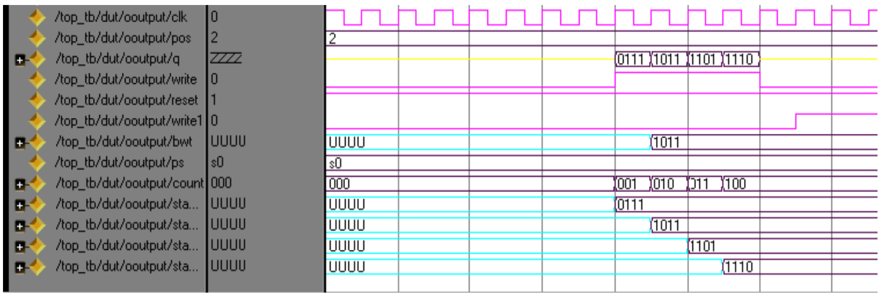
/top_tb/dut/ooutput/clk 0
/top_tb/dut/ooutput/pos 2
/top_tb/dut/ooutput/q ZZZZ
/top_tb/dut/ooutput/write 0
/top_tb/dut/ooutput/reset 1
/top_tb/dut/ooutput/write1 0
/top_tb/dut/ooutput/bwt UUUU
/top_tb/dut/ooutput/ps s0
/top_tb/dut/ooutput/count 000
/top_tb/dut/ooutput/sta... UUUU
/top_tb/dut/ooutput/sta... UUUU
/top_tb/dut/ooutput/sta... UUUU
/top_tb/dut/ooutput/sta... UUUU
2
0111 1011 1101 1110
UUUU 1011
s0
000 001 010 011 100
UUUU 0111
UUUU 1011
UUUU 1101
UUUU 1110

Fig. 3. VHDL Simulation output of Reversed BWT operation

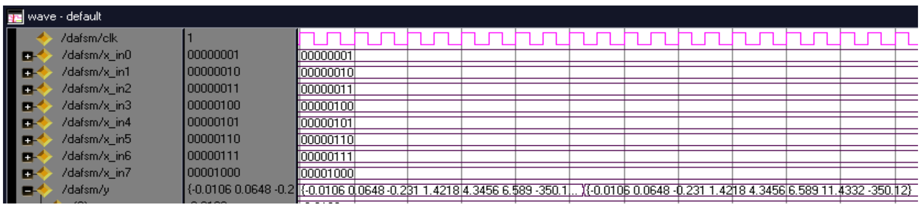

Fig. 4. VHDL Simulation output of a low pass filter with Daubechies-4 wavelet coefficients using distributed arithmetic architecture

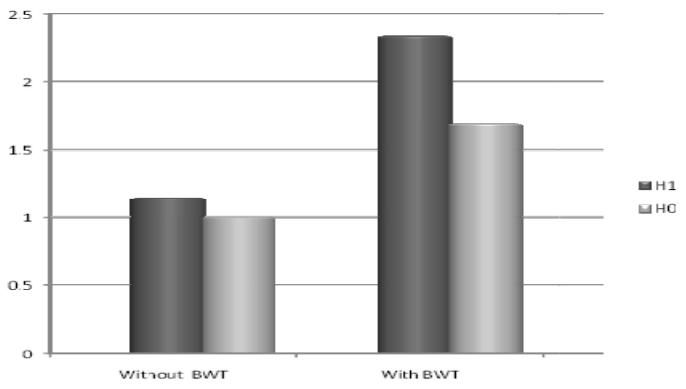

Fig. 5. Comparison of compression ratios of look-up tables with and without BWT for Daubechies-4 high pass(H1) and low pass(H0) filter coefficients

5 Conclusion

The synthesis report generated by Xilinx ISE 9.1 revealed that the time needed for executing the BWT block is only 5.00 seconds. This guaranteed that addition of BWT block will not affect much delay in the entire operation. This method is tested in the implementation of DWT using DB4 wavelet and obtained a compression ratio of 2.3:1. Thus this methodology of adding BWT block to any distributed arithmetic computation of filters with larger coefficients guarantees a reduction in the memory needed for look-up tables.

References

1. White, S.A.: Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review. IEEE ASSP MAGAZINE (July 1989)
2. Rao, R.M., Bopardikar, A.S.: Wavelet Transforms: Introduction to Theory and Applications. Addison-Wesley (2000)
3. Burrows, M., Wheeler, D.J.: A Block-sorting Lossless Data Compression Algorithm. SRC Research Report (May 1994)
4. Al-Haj, A. M.: An FPGA-based Parallel Distributed Arithmetic Implementation of the 1D Discrete Wavelet Transform. International Journal of Computing and Informatics (Informatica) 29(2) (2005)
5. Bonny, T., Henkel, J.: Efficient Code Compression for Embedded Processors. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 16(12), 1696–1707 (2008)