IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

VLSI Design of Area Efficient High Performance SPMV Accelerator using VBW-CBQCSR Scheme

N. Narasimharao[1], A. Mallaiah[2]

M.Tech (Embedded Systems) Gudlavalleru Engineering College[1]

Associate Professor, Dept. of ECE, Gudlavalleru Engineering College[2]

Abstract: This project presents a high performance sparse matrix-vector multiplication (SpMV) accelerator on the field-programming gate array (FPGA). By exploiting a hardware-friendly storage theme, named as Variable-Bit-Width Coordinate Block similar Compressed distributed Row, the redundant computation and memory accesses are often reduced greatly through the nested block compression and variable-bit-width column-index secret writing schemes. Supported the planned compression theme, a deeply-pipelined SpMV accelerator is enforced on a Xilinx Virtex XC7VX485T FPGA platform, which may handle distributed matrices with absolute size and poorness pattern. Dadda number is one in all the quickest number utilized in several data-processing processors to perform quick arithmetic functions. The most elements of Dadda number area unit full adder and half adder modules that area unit used for playacting the reduction of the partial Merchandise. A full adder is used because the key part within the style of those multipliers to cut back space and delay. In this work, 8-bit DADDA number victimization VBW-CBQCSR theme is enforced. Experimental results show that the projected style will gain less delay and space is additionally less and reduces the power consumption as compared to the previous works.

Keywords: Sparse Matrix Vector Multiplication (SPMV), FPGA, accelerator, VBW-CBQCSR.

I. INTRODUCTION

It high performance sparse matrix-vector multiplication (SPMV) accelerator on the field-programming gate array (FPGA). By exploiting a hardware-friendly storage scheme, named as Variable-Bit-Width Coordinate Block Quasi Compressed Sparse Row, the redundant computation and memory accesses can be reduced greatly through the nested block compression and variable-bit-width column-index encoding schemes. Based on the proposed compression scheme, a deeply-pipelined SpMV accelerator is implemented on a Xilinx Virtex XC7VX485T FPGA platform, which can handle sparse matrices with arbitrary size and sparsity pattern.

Sparse matrix-vector multiplication (SpMV) is one of the most essential kernels in scientific computing, such as sparse linear solvers, image processing, Circuit analysis and so on.The customizable feature of FPGAs can be used to design the application-specific memory structures and processing elements to match the compression schemes, which can increase the utilization of memory bandwidth.

A new VBW-CBQCSR scheme is proposed to exploit the bit capacity of FPGA. A multiplier is one of the key hardware blocks in most digital and high performance systems such as FIR filters, digital signal processors and microprocessors etc. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following- high speed, low power consumption, regularity of layout and hence less area or even combination of them in multiplier [14]. Generally all multiplication methods share the same basic procedure - addition of a number of partial products. The simple methods are easy to implement, but the more complex methods are needed to obtain the fastest possible speed.

Luigi Dadda, the computer scientist has proposed the Dadda algorithm for multiplication during 1965. It is slightly faster and requires fewer gates. Different types of schemes are used in parallel multiplier. The Dadda scheme is one of the parallel multiplier schemes that essentially minimize the number of adder stages required to perform the summation of partial products. This is achieved by using full and half adders to reduce the number of rows in the matrix number of bits at each summation stage. Even though the Dadda multiplication has regular and less complex structure, the process is slower due to serial multiplication process. Further, Dadda multiplier is less expensive compared to that of Wallace tree multiplier.

II. RELATED WORK

Sparse matrix-vector multiplication performs the operationy=Ax, where A is a large and sparse matrix, and xandy are dense vectors. In order to achieve higher performance, it is required that designing the proper compression scheme of the sparse matrix to fully utilize the underlying system architecture. There have been many compression schemes proposed in the literature, such as

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Coordinate (COO), Compressed Sparse Row (CSR), Compressed Sparse Column (CSC), ELLPACK-ITPACK (ELLPACK), and so on. There has been a substantial body of works to implement the SpMV on FPGA, which can be divided into two categories, the novel compression schemes and pipelined architecture based on the conventional compression schemes.

From the perspective of the novel compression schemes, due to the limited capacity of the on-chip Block RAM, block is one of the most used ways to compress the sparse matrices. Given the disposal of the sub matrices, the block scheme can be divided into two groups. In the first group, such as BCSR and Row Blocked CSR, the sub matrices are considered as dense matrices, and the index data is reduced by only recording the indices of the nonzero blocks, instead of each nonzero element. However, the excessive zero padding to construct the dense sub matrices degrade the performance. In the second group, the sub matrices are taken as sparse matrices, and compressed with specific scheme to decrease the redundant computation and memory requirement by reducing the zero padding. However, the overhead of the word-level-encoded index data of each nonzero element limits the performance improvement. As the works in, the overhead can be reduced by replacing the indices with bitmap, and the indices are retrieved through the decoding before the computing. However, the performance of these works is restricted by the idle cycles in the index decoding and the zero fillings in the bitmap

III. PROPOSED SYSTEM

In order to fully utilize the bit capacity of FPGA to improve the performance of SpMV, a hardware-friendly compression scheme, named as VBW-CBQCSR, is proposed. The VBW-CBQCSR scheme consists of two parts, CBQCSR and VBW, which are used to compress the sparse matrix and the column indices of the non zero elements, respectively.

The CBQCSR scheme partitions the sparse matrix in the 2D uniform way, and stores the sub matrices in a two-level storage format. For the first level storage scheme, a quasi-COO format is used to store the indices of the nonzero sub matrices. The row and column indices of the sub matrices are stored in array brow and bcol in a row-wise order. The array bval is used to store the start position of each sub matrix in the second level storage scheme.

For the second-level storage scheme, A Quasi compressed sparse row (QCSR) scheme is proposed to reduce the memory access and redundant computation. The QCSR scheme contains three 1D arrays: Val, col and EOR. The values and column indices of the nonzero elements are stored in row-wise order in the Val array and col array in a one-to-one manner. Instead of the row ptrarray in the conventional CSR scheme, the EOR(end-of-row) flags are introduced to mark the termination of each row of the nonzero sub matrices. The EOR flags are stored in the EOR array. When the value of EOR[i]is one, the corresponding Val[i]and col[i]are the last items of one row. Through the EOR flag, the parallelism across multiple rows can be exploited to improve the performance. The main idea of the VBW part is to make use of the variable bit width encoding scheme to reduce the number of bits required to store the column indices.

IV. OVERALL ARCHITECTURE OF THE SPMV ACCELERATOR

Based on the proposed VBW-CBQCSR scheme, a deeply-pipelined SpMV accelerator is implemented on a self-designed FPGA platform with one Xilinx Virtex-7FPGA and three external DRAM Memory modules, as shown in below Figure 1. The sparse matrix and vector x are all stored in the external DRAM Memory modules.

The processing elements (PE[1],..., PE[n]) access the data through the Customized Memory Interface and execute the SpMV on different block rows in parallel. When the computation of one block row is finished, the results of the vectory are written back to the external DRAM Memory modules.

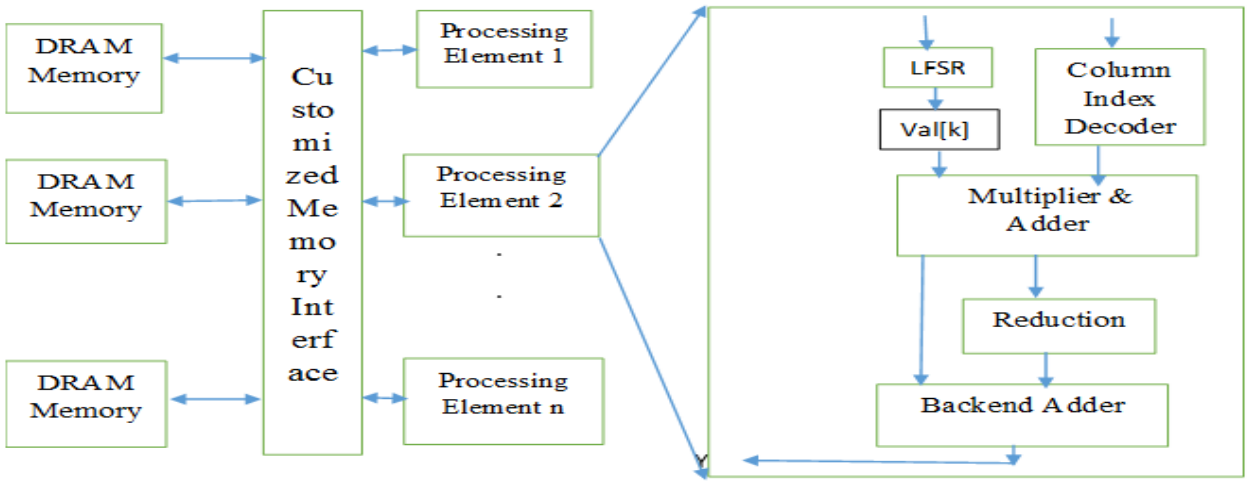

Fig 1: Overall Architecture of the SPMV accelerator using VBW-CBQCSR

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Each PE contains four main components, Column Index Decoder, Dadda Multiplier adder, Reduction and Backend Adder. The Column Index Decoder moduleretrieveskcolumn indices in parallel, which are used to access the data of the vectorx. The kpairs ofx[i]andval[i]are multiplied and accumulated by the Dadda Multiplier Adder module without the limitation of the same row.

The partial sums withpre_EOR[k] = 0are fed into the Reduction module,wherepre_EOR[k]stands for theEORflag of the lastk-th partial sum. The Reduction module accumulates the partial sums of the same row. Others can be directly fed into the Backend Adder module. The BackendAdder module adopts a multi-bank architecture to accumulate the partial sums ofthe block being processed with the partial sums out of the same rows in theprevious blocks in parallel. After the computation of one block row isfinished, theresults of the vectoryare written back to the external DRAM Memory modules.

V. DADDA MULTIPLIER

The Dadda multiplier is slightly faster (for all operand sizes) and requires fewer gates (for all but the smallest operand sizes) than array multiplier [15].
Dadda multiplication comprises 3 steps:

1. Multiply (i.e., AND) each bit of one of the arguments, by each bit of the other, yielding N^2 results. Depending on position of the multiplied bits, the wires carry different weights.
2. Reduce the number of partial products to two layers of full and half adders.
3. Group the wires in two numbers, and add them with a conventional adder.

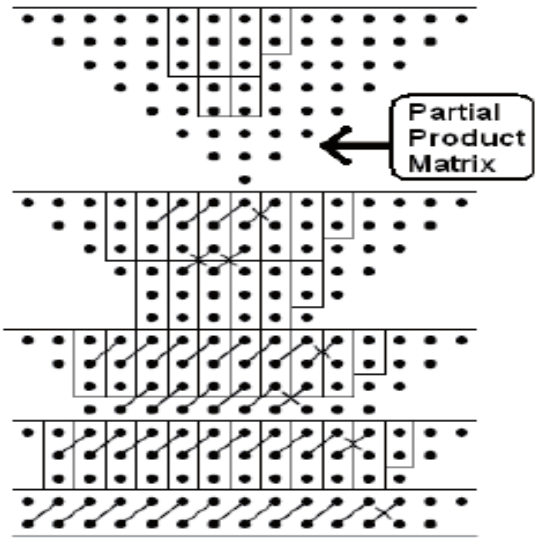

Fig 2: DOT Diagram of 8*8 DADDA Multiplier

Dadda multipliers perform few reductions only when compared to Wallace multiplier. Because of this, Dadda multipliers have less expensive reduction phase, but the numbers may be a few bits longer, thus requiring slightly bigger adders.

VI. EXPERIMENTAL RESULTS

Simulation Results:

The Proposed deeply-pipelined FPGA based SpMV accelerator using VBW-CBQCSR scheme is designed using Xilinx ISE 14.2 Tool and modeled in Verilog HDL. The simulation Results and RTL, Technology schematic diagrams are shown below figures 3, 4, 5 and 6.

Top Module of SpMV Accelerator:

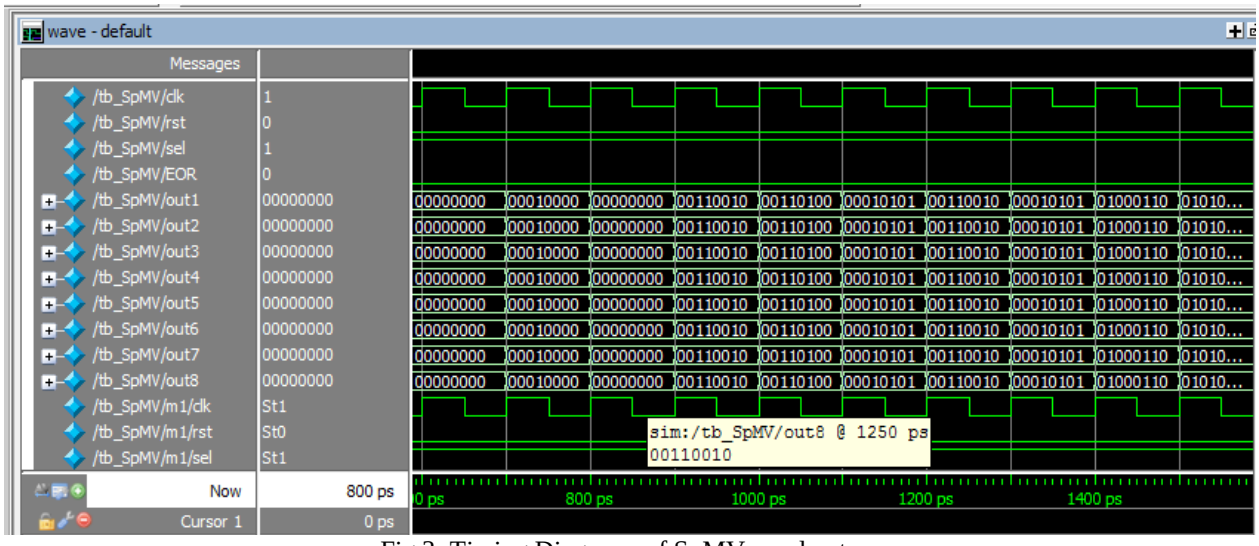

Fig 3: Timing Diagrams of SpMV accelerator

These are the Timing diagrams of the Top Module of the SpMV Accelerator using Variable Bit Width Coordinate Block Quasi Compressed Sparse Row (VBW-CBQCSR).

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

SpMV using Dadda Multiplier:

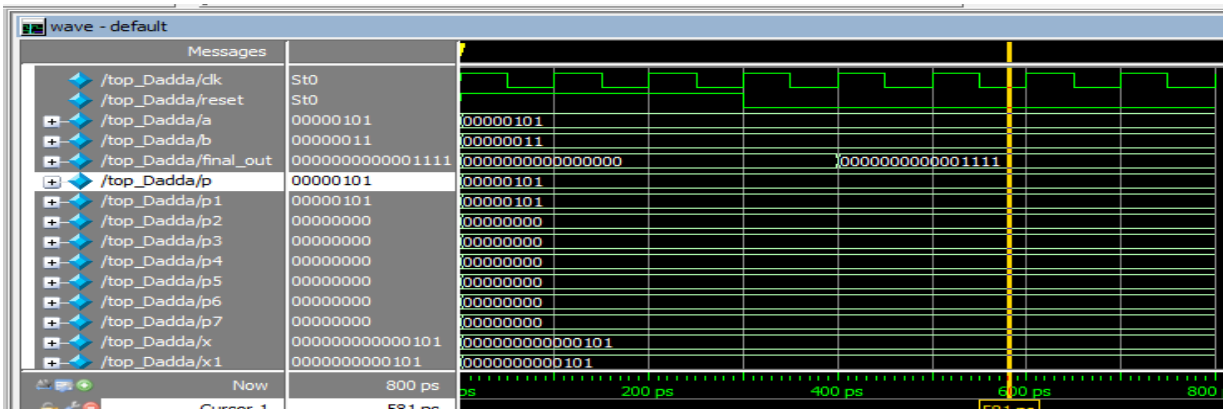

Fig 4: Timing Diagrams of SpMV using Dadda Multiplier

These are the Timing diagrams of the Top Module of the SpMV Accelerator using Dadda Multiplier using VBW-CBQCSR.

Synthesis Results:
RTL schematic:

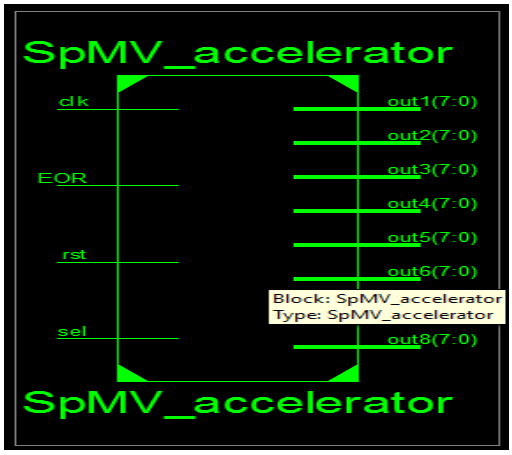

Fig 5: RTL schematic Diagram for SpMV accelerator.

This is the RTL schematic Diagram for the SpMV accelerator and technology schematic Diagram is also placed here. These are explanation of which input we are going to give and which output we are going to get. SpMV accelerator has inputs are 4 general inputs and the 8 outputs.

Technology Schematic:

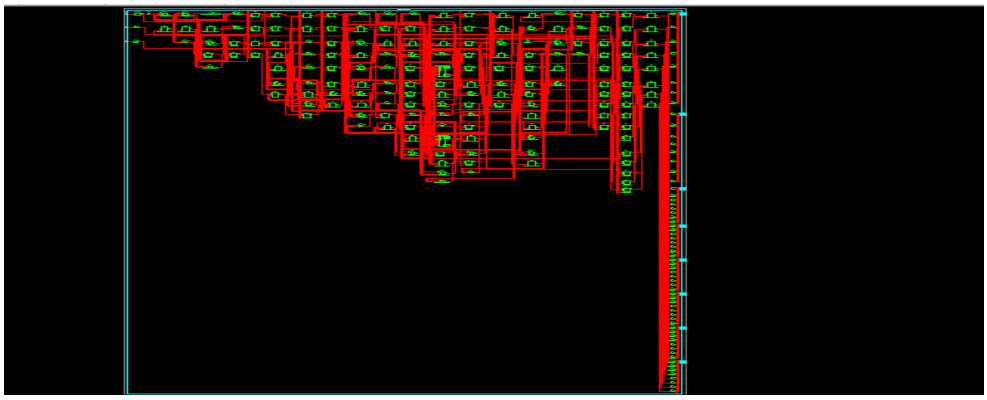

Fig 6: Technology Schematic Diagram for SpMV accelerator.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Synthesis Report:

For the performance evaluation and comparison, the SpMV accelerator is implemented on an Xilinx Virtex-7 XC7VX485T FPGA platform. The design is described in RTL with Verilog HDL and synthesized with ISE 14.2. Our design is determined by several parameters, i.e., the block size b, the 8 number of PE's, the number of the frontend multipliers 2, and the pipeline depth of the floating point adder. The synthesis results reported by the Xilinx ISEare given in Table I.

Table I. Synthesis Results of the Present Work SpMV accelerator

	Estimated Values			Present Work Values		
Logic utilization	**used**	**Available**	**utilization**	**used**	**Available**	**utilization**
Number of slices	59	4656	1%	26	4656	0%
Number of slice Flip Flops	76	9312	0%	48	9312	0%
Number of 4 input LUTS	103	9312	1%	29	9312	0%
Number of bonded IOBS	67	232	28%	66	232	28%
Number of GCLKs	1	24	4%	1	24	4%

	Estimated Results	**Present Work Results**
Area	1.3%(utilization)	0.722(utilization)
Delay	5.418ns	3.447ns

VII. CONCLUSION

This paper presents a deeply-pipe lined SpMV accelerator on FPG Ausinga Variable Bit Width Coordinate Block Quasi Compressed Sparse Row (VBW-CBQCSR) scheme. By employing nested block compression and variable-bit-width column index compression, the compression scheme can greatly reduce their dun ant computation and memory accesses. Based on this scheme, a SpMV acceleratory simpleminded on an Xilinx VirtexXC7VX485TFPG A platform, which can handle sparse matrices with arbitrary size and sparsity pattern. The accelerator can exploit the parallel is macros multiplier owsto improve the performance. The most elements of 8-bit Dadda number area unit full adder and half adder modules that area unit used for playacting the reduction of the partial merchandise. Experimental results show that the projected style will gain less delay and space is additionally less and reduces the power consumption as compared to the previous works.

REFERENCES

[1] S. Kestur, J. D. Davis, and E. S. Chung, "Towards a universal FPGA matrix-vector multiplication architecture," in Proc. FCCM, 2012, pp.9–16.

[2] Paul Grigoras, Pavel Burovskiy, Eddie Hung and wayneLuk "Improving SpMV Performance on FPGAs through lossless Nonzero compression"Departments of computing Imperial college London,2014.

[3] G. Kuzmanov and W. M. van Oijen, "Floating-point matrix multiplication in a polymorphic processor," in In ICFPT International Conference Field-Programmable Technology, 2007, pp. 249–252.

[4] S. Sun, M. Monga, P. H. Jones, and J. Zambreno, "An I/O Bandwidth-Sensitive Sparse Matrix-Vector Multiplication Engine on FPGAs," Circuits and Systems I: Regular Papers, IEEE Transactions on, vol. 59,no. 1, pp. 113–123, 2012.

[5] J. Fowers, K. Ovtcharov, K. Strauss, E. S. Chung, and G. Stitt, "A high memory bandwidth FPGA accelerator for sparse matrix-vector multiplication", 2014.

[6] D. Gregg, C. Mc Sweeney, C. McElroy, F. Connor, S. McGettrick,D. Moloney, and D. Geraghty, "FPGA based sparse matrix vector multiplication using commodity DRAM memory," in Proc. FPL. IEEE,2007, pp. 786–791.

[7] M. DeLorimier and A. DeHon, "Floating-point sparse matrixvector multiply for FPGAs," in FPGA, H. Schmit and S. J. +E.Wilton, Eds. ACM, 2005, pp. 75–85.

[8] Y. El-Kurdi, W. J. Gross, and D. Giannacopoulos, "Sparse matrixvector multiplication for finite element method matrices on FPGAs," in FCCM. IEEE Computer Society, 2006, pp. 293–294.

[9] L. Zhuo and V. K. Prasanna, "Sparse matrix-vector multiplication on FPGAs," in FPGA, H. Schmit and S. J. E.Wilton, Eds. ACM, 2005, pp. 63–74.

[10] J. Sun, G. D. Peterson, and O. O. Storaasli, "Sparse matrix-vector multiplication design on FPGAs," in FCCM, K. L. Pocek and D. A.Buell, Eds. IEEE Computer Society, 2007, pp. 349–352.