HDL Based Design for High Bandwidth Application

M. Mohammed Kasim [1] K. Nagarajan [2] and P. Kumar [3]
[1,2,3.] Assistant Professor, Dept of ECE,
Nehru Institute of Engineering and Technology, Coimbatore – 641 105. Tamilnadu. India

Abstract— The objective was to configure a architecture for high bandwidth applications such as multi-media and wireless communications. The existing problem in high bandwidth applications are inflexibility and inefficiency .The loop level parallelism, reconfigurable controller, memory bandwidth were addressed as critical issues in the proposed architecture. The architecture can be used for different word length operations. This architecture can reduce the inflexibility and inefficiency problems.

I. INTRODUCTION

High bandwidth applications require high computing power, flexibility, and scalability. The ASIC solution gives high computing power requirement, but is inflexible and not scalable. The general purpose microprocessors or DSP are flexible, but insufficient in computing power. Since early 1990s, reconfigurable architectures compromise be fulfill the two extreme solutions, and been applied for high bandwidth applications such as multimedia [1].

Critical loop operations in multimedia and wireless communications usually consume a good portion of the total execution cycles. Therefore, the key issue in implementing multimedia or wireless algorithms onto a reconfigurable architecture is to map critical loops into processing elements optimally to meet the computing need [2].

A reconfigurable architecture has evolved from fine grained to coarse-grained architecture. This concerns only coarse grained architectures due to some major advantages such as efficient area high performance and low power [3]. Existing coarse-grained architectures can be categorized into two groups, data path-oriented and instruction oriented architectures, based on the type of execution performed by processing elements.

A processing element for a data path-oriented architecture executes only one type of operation once it is configured, and required data flow is constructed by routing mesh structured processing elements. To implement the LLP on a data path-oriented architecture, the body of the loop is replicated on mesh and multiple iterations are executed concurrently in a pipeline manner [4]. And also it does not lead to high resource utilization when I/Os from/to processing elements are limited. Data flow does not fit into a given mesh topology. Also the degree of parallelism in LLP is limited [5].

And in the same time processing element of an instruction-oriented architecture performs sequence of operations, which are defined by instructions, micro-code and/or control signals. Instructions are stored in a configuration memory and fetched by a controller to control the associated processing element [7]. A processing element can execute the entire body of a loop, but LLP is simply to assign multiple processing elements running concurrently. So instruction oriented architecture leads to high resource utilization and more suitable than the former type of architectures for multimedia and wireless communication.Although the instruction-oriented architectures are suitable for multimedia and wireless communication, there are several problems in existing architectures. To moderates these problems we proposed a new instruction-oriented reconfigurable architecture.

Some critical loop operations in multimedia and wireless communications usually consume a good portion of the total execution cycles. Therefore, the key issue in implementing multimedia or wireless algorithms onto a reconfigurable architecture is to map critical loops into processing elements optimally to meet the computing need.

II. RECONFIGURABLE COMPUTING IN WIRELESS

This paper proposes about the computational demand of wireless communication [8], a new approach was implemented for new standards. Reconfigurable processing can meet the needs for wireless base station design while providing the programmability to allow upgrades as standards.

A reconfigurable communication processor was used here to meet the computational demand of wireless industry. But the architecture cannot be used for wide band width. The instruction memory is too small and Sub word parallelism is used in a limited portion. Also this architecture has complex interconnection.

Sub-word parallelsim (SWP)

Sub-word parallelism is used to increase to the parallelism by partitioning the data path into sub word [9]. The multiple sub word can be processed concurrently to increase the computational efficiency. This concept was effectively used in high band width application. It deals parallelism at data level. The sub-word parallelism increases parallelism at the data element path by partitioning the data path of the processor.

The Bit adder can be partition. The figure 1 shown below was the best example for partitioning.

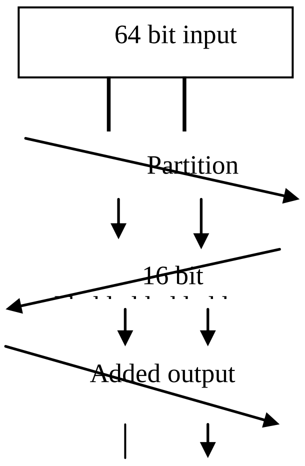

Fig.1 SWP 16 bit adder

Loop level parallelism

The loop level parallelism (LLP) method was proposed [10]. The execution of the loop was considered .The execution time must be minimized. Parallel processor systems that have been built so far can execute in parallel only singly nested parallel loops. However, it is crucial to be able to exploit multidimensional parallelism which occurs in multiply nested parallel loops. Processor assignment algorithms are presented for simple and complex nested parallel loops. These processor assignment schemes can be used by the compiler to perform static processor which allocated to multiply nested parallel loops.

The schemes for statically allocating processors to several nested parallel loops such that parallel execution time is minimized. This method was implemented in the compiler or it can perform the assignment of processors to loops in a dynamic way.

In parallel computers the parallelism was done in single loop. Parallelism can be better utilized if a limited number of processors are allocated to more than one loop. The execution operation is concurrently done with processing element. The schemes for statically allocating processors to several nested parallel loops such that parallel execution time is minimized. This method was implemented in the compiler or it can perform the assignment of processors to loops in a dynamic way.

Wide memory bandwidth in reconfigurable accelerator:

This is instruction oriented architecture. Programming storage element (PSE) was mainly used here. The source utilization is high. The paper considered for memory band width operations. The memory access requirement is high in certain time. So the processing element with limited band width should wait until the necessary data is available from the memory. The processing elements exceed the available memory I/O. This is limited bandwidth operation and not suitable for wide bandwidth functions [11].

III. Reconfigurable Architecture of High Band Width Applications

The present work proposes new architecture for high bandwidth applications. This resolves the problems like inflexibility and inefficiency in the high bandwidth applications. Wide memory bandwidth, loop level parallelism was kept as main issues in reconfiguring the architecture. The reconfigured architecture can increase the performance of high bandwidth applications.
The architecture was designed by those blocks given below. The overall architecture is shown in figure 2. It has an array of processing elements slice (PES), which may be concatenation for scalability. One PES consists of local memories, XBSN (cross bar switch network), 16 PEM (processing elements and multiplier) and a RC (reconfigurable controller). A PES is a basic block for execution of multiple iterations of a loop in parallel. The number of iterations of a loop which can be executed concurrently on a PES depends on the type of the operations.

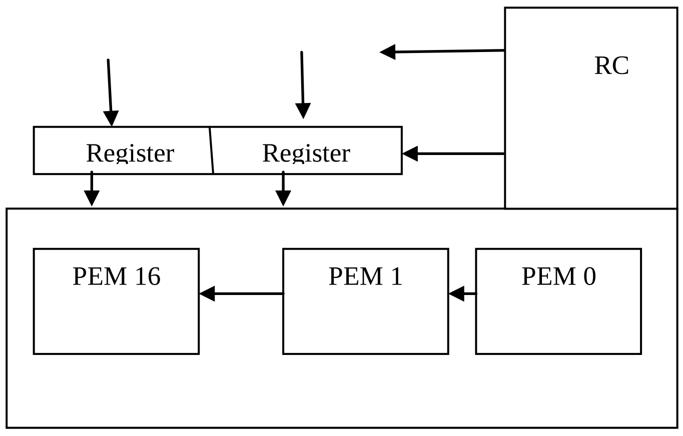

Fig 2: Reconfigurable architecture of high band width applications

IV. Processing Element Slice

This basic unit for loop level execution. It has two 16 Kb memories for the demand of memory access. The local memory has wide input and output port. Thus it accesses the entire data in a single clock cycle. Like other instruction-oriented architectures the proposed architecture has a 1-d array of PESs. A PES is the basic processing unit for the LLP execution

To meet the peak demand on memory access, the PES has two 16 KB (256X512) column less memories. The local memory has a 256-bit wide input/output port without a column decoder, and thus accesses an entire row data on a single clock cycle. Sense amplifiers are used as a register file, which loads 64 8-bit data concurrently from the memory.To support various types of memory accesses necessary for multimedia and wireless communication applications and to minimize the communication overhead among PEs and each PES includes an XBSN.

Thus it accesses the entire data in a single clock cycle. Like other instruction-oriented architectures the proposed architecture has a 1-d array of PESs. A PES is the basic processing unit for the LLP execution.

A. *Memory:*

This is the important part of processing element. The basic memory details and the memory used for this architecture was given below.RAM (random access memory) is the place where the application programs, and data in current use are kept so that they can be quickly reached . RAM is much faster to read from and write to than the other kinds of storage in a computer, the hard disk, floppy disk, and CD-ROM. However, the data in RAM stays there only as long as your computer is running. When you turn the computer off, RAM loses its data. When you turn your computer on again, your operating system and other files are once again loaded into RAM. RAM can be compared to a person's short-term memory and the hard disk to the long-term memory. The short-term memory focuses on work at hand, but can only keep so many facts in view at one time. If short-term memory fills up, your brain sometimes is able to refresh it from facts stored in long-term memory. A computer also works this way. If RAM fills up, the processor needs to continually go to the hard disk to overlay old data in RAM with new, slowing down the computer's operation. Unlike the hard disk which can become completely full of data so that it will accept any more, RAM never runs out of memory. It keeps operating, but much more slowly than you may want it to.

During the peak cycles, processing elements withal limited memory bandwidth should wait until the necessary data is available from the memory. Therefore, a wide memory bandwidth is a critical design issue to achieve a high degree parallelism for the LLP, which is difficult for high performance for instruction oriented architectures. The existing architecture was not suitable for wide memory bandwidth .

Here the Local memories store the input and output data streams to read from or to write onto the host processor and PEMs. To meet the peak demand on memory access, the PES has two 16 kB (256x512) column less memories. The local memory has a 256-bit wide input/output port without a column decoder, and thus it accesses an entire row data on a single clock cycle.

B. *Cross bar switches:*

Crossbar switches are widely used today in a variety of applications including network switching, parallel computing and various telecommunications applications. To avoid the communication over head problem, the cross bar switches are used. It avoids the over head communication between program elements and program element slices.

An XBSN includes two 32x32 8-bit crossbar switches, so any 8-bit word of 32 operands (fetched from a memory) can be available to each operand register. In other words, 64 8-bit operands are fetched from the two memories on a clock cycle and available to the two operand registers, in turn, the 16 PEMs.

A crossbar switch, also known as a cross point switch, is defined as a switch with n input lines and n output lines (n port switch). The switch has n2 intersections, called cross points, where an input line and output line can be electrically connected .As can be seen from Figure, the number of cross points grows as the square of the number of lines into the switch. If we assume that a switch port does not connect to itself, the number of cross points needed is given by n(n-1)/2.For n=32, there are 496 cross point connections. Splitting the crossbar switch into several smaller switches and interconnecting them can dramatically reduce the number of cross points. This technique is called space division switching. There is a penalty inherent in this technique known as blocking.

The switch has 32 inputs that connect to any or all 32 outputs via the 32 x 32 switch matrix. The switch matrix is constructed with thirty-two 32-input multiplexers. Each output supports individual tri state control. The MUX select lines (switch interconnects) and tristate control are configured using a set of double buffered configuration registers. The LOAD Registers are loaded for each port individually by asserting the LOAD and CS signals. The Output Address lines are decoded to select the port's LOAD register, and the input address lines are latched along with the TRI signal. The latched input address lines drive the port's MUX select lines, and TRI drives the port's tristate control. After the LOAD registers have been configured, the CNFG and CS signals are asserted, simultaneously configuring all 32 ports. This double buffering scheme prevents any data from being lost while the switch interconnects are updated.

C. *Reconfigurable controller:*

The controller generates control signals for local memories, cross bar switches, processing element and multiplier. The finite state machine was used here to reduce hardware complexity. Controller generates control signals for local memories, the XBSN, and PEMs for each instruction pipeline stage.
This employs a reconfigurable controller which is a fine grained LUT (look up table) block like an FPGA. Unless previous memory based controllers, the proposed controller implements for a finite state machine (FSM) to generate control signals [11]. Since a combinational logic in FSM can be minimized through logic optimization technique, more functionality than the memory based controller can be provided the given area.Also controller can be shared across multiple processing elements through flexible implementation of a FSM, which reduces the overall hardware complexity.

D. Processing Element and Multiplier (PEM)

The PEM consist of multiplier with 9 bit and two processing elements. The processing element multiplier is responsible for a single multiplication or two additions. The processing element has three 8bit ALU's and 8bit register, temporary registers to store the output of arithmetic and logic unit. This unit is responsible for addition, subtraction. It works as a carry select adder.

The arithmetic-logic unit (ALU) performs all arithmetic operations (addition, subtraction, multiplication, and division) and logic operations. Logic operations test various conditions encountered during processing and allow for different actions to be taken based on the results. The data required to perform the arithmetic and logical functions are inputs from the designated CPU registers and operands. The ALU relies on basic items to perform its operations. These include number systems, data routing circuits (adders/ subtracters), timing, instructions, operands, and registers.

V. SUB WORD PARALLELISM WITH FLEXIBLE WORD LENGTH SUPPORT:

This supports flexible word-length operations-addition/ subtraction, shift and multiplication. For addition/ subtraction operations, multiple 8-bit PEs can simply concatenate to construct a higher precision, where each PE configures as a carry selection adder to minimize the critical path delay of a long word-length addition/subtraction. For shifting operation, the XBSN provides various word-length parallel shift operations-8-bit, 16-bit and 32-bit. In the XBSN, scramble multiplexer performs bit scrambling operation so that the cross bar switch can arrange the bit position according to the shift amount, and multiple shifted data can be obtained by de-scrambling the output of crossbar switch.

A new method is proposed to provide flexible word-length multiplications. Since any MAC(multiplication and accumulation) operations can be expressed with low-precision MACs, This provides various types of MACs using 8X8 atomic MAC units , PEMs, the XBSN divides multiplicands into 8-bit.

VI. CONCLUSION

Architecture of high band width applications can be reconfigured. This can increases the performance of the high band width application architecture. The memory was designed and the processing element multiplier (PEM) part was simulated. The processing element multiplier has two processing elements which can do arithmetic operations parallel.

REFERENCES

[1] R. Hartenstein, "A Decade of Reconfigurable Computing: a Visionary Retrospective", Design, Automation and Test in Europe, Conference and Exhibition Proceedings, 2001.
[2] C.D. Polychronopoulous, et al., "Utilizing multidimensional loop parallelism on large-scale parallel processor systems", IEEE Transactions on Computers, Vol. C-38, No. 9, Septembe1989.
[3] T. Miyamori and K. Olukotun, "A Quantative Analysis ofReconfigurable Comprocessors forMultimedia Applications", IEEESymposium on FPGA for Custom Computing MachineApril 1998, pp. 2- 11.
[4] E. Mirsky and A. DeHon, "MATRIX: a reconfigurable computingarchitecture with configurable instruction distribution and deployable resources", Proceedings. IEEE Symposium on FPGA for Custom Computing Machines 1996.
[5] H. Singh, "MorphoSys: An Integrated Reconfigurable System for Data-Poarallel and Computation-Intensive Applications", IEEETrans. On Computers, Vol. 49, No. 5, 2000.
[6] V. Baumagarten, "PACT XPP – a self-reconfigurable data processing architecture", Journal of Supercomputing, 2003, pp. 167-184.
[7] M.B. Taylor, "The Raw microprocessor: a computational fabric for software circuits and general purpose programs", IEEE Micro,Vol. 22, Issue 2, March-April 2002, pp. 25-35.
[8] B. Salefski and L. Caglar, "Re-Configurable Computing in Wireless", Proceedings of Design Automation Conference, 2001, pp. 178-183.
[9] J. Fridman, "Sub-word Parallelism in Digital Signal Processing",IEEE Signal Processing Magazine, March 2000.
[10] J. Leijten, J. Huisken and E. Waterlander, A. V. Wel, "AVISPA: AMassively Parallel Reconfigurable Accelerator", Proceedings ofInternational Symposium on System-on-Chip, 2003, pp. 165-168.
[11] C.D. Polychronopoulous, "Utilizing multidimensional loop parallelism on large-scale parallel processor systems", IEEE Transactions on Computers, Vol. C-38, No. 9, Septembe1989.