Design of Reconfigurable Floating Point Arithmetic and Logic Unit

Mahadevan.V.S\textsuperscript{1}, Prasanna.V\textsuperscript{2}, Sathiya. S\textsuperscript{3}, Suganthi. I\textsuperscript{4}, Vandanapriya. A\textsuperscript{5}
Assistant Professor\textsuperscript{1}, Student\textsuperscript{2,3,4,5}
Department of Electronics and Communication Engineering
Achariya College of Engineering Technology, Puducherry, India

Abstract:
Reconfigurable Computing has grown to become an important and large field of research. This paper describes the implementation of a reconfigurable ALU that combines 32-bit single precision floating point adder, 64-bit double precision floating point adder, and integer ALU (arithmetic logic unit) into a single unit on FPGA. Reconfigurable ALU can perform both integer operations and floating point additions. Simply extending the operand width and adding a few switches enable the unit to be reconfigured. The proposed architecture of this paper analysis the logic size, area and power consumption using Xilinx 14.2.

Keywords: floating point, reconfigurable, single precision, double precision, ALU.

I. INTRODUCTION

Floating-point arithmetic is very useful in many applications. However, non-numerical applications usually have very few floating-point computations, and as a result, the floating-point resources are mostly in idle mode. During this idle mode, the floating-point still consume power and the die area is wasted. Therefore, reconfigurable ALU can be used in microprocessor to reduce power consumption and avoid waste die area. VHDL includes facilities for describing logical structure and function of digital systems at a number of levels of abstraction, from system level down to the gate level. It is intended, among other things, as a modeling language for specification and simulation. We can also use it for hardware synthesis if we restrict ourselves to a subset that can be automatically translated into hardware. VHDL arose out of the United States government’s Very High Speed Integrated Circuits (VHSIC) program. In the course of this program, it became clear that there was a need for a standard language for describing the structure and function of integrated circuits (ICs). Hence the VHSIC Hardware Description Language (VHDL) was developed. It was subsequently developed further under the auspices of the Institute of Electrical and Electronic Engineers and adopted in the form of the IEEE Standard 1076, Standard VHDL Language Reference Manual, in 1987 with the technological progress of programmable devices and programming methodologies, reconfigurable devices become more and more interesting and important. In this paper, such technology is applied for designing reconfigurable ALU. ALU is the heart of microprocessors as it executes binary instructions of a program. Typical issues for VLSI designers are to reduce the area of the chip and increase its performance for computational applications like video compression, graphics, gaming consoles etc. But the development of portable devices and palm held devices, has forced the designers to optimize the power consumption of the device while still meeting the computational requirements. Reconfigurable computing provides the flexibility in arriving at the problem specific architectures which helps in improving the performance due to custom approach. We implement the reconfigurable architecture by exploring the regularity of the adder architectures with minimum additional multiplexers. Here, re-configurability has been achieved between the adder variants. The growing design complexity has attracted the designs with reconfigurable fabrics, where adaptable fabrics are utilized to solve the computational problems.

II. DESIGN OF PROPOSED ALU

In the proposed system to design the floating point adder in 64 bit double precision. First we are converting the floating numbers in the form of sign bit, exponent bits and mantissa bits then pass to the floating point ALU design. In the logic unit is contain basic logic function. And the arithmetic unit is containing the adder and subtract or unit. The design of arithmetic unit is given below.

Figure 1. Floating adder design
From the adder design, to use the swapping logic for which one need shift for arithmetic process and to use the comparator to set the A is greater or lesser. If A is greater means to get the output of comparator is high else low. The signal is pass to swapping module and the difference of 2 inputs exponent is used to set the how much shift is need in shifter module. Then adder or subtract the input data. The hardware implementation of this arithmetic for floating point is a complicated operation due to the normalization requirements.

### RTL VIEW OF 64 BIT ALU

Figure 2. RTL View

The design of proposed design adder is the input data is converted into the following formats:

- S: Sign bit
- M: Mantissa bits
- E: Exponent bits

For 32 bit floating point number the bits are 1 bit for sign, 8 bit for exponent and 23 bit for mantissa. For 64 bit floating point number means 1 bit for sign, 11 bit for exponent and 53 bit for mantissa. Floating point addition is one of the difficult units in the floating point arithmetic operations. Addition and the related operation of subtraction is the most basic arithmetic operation. The logic unit contain the basic gates of AND, OR, NOR, NAND, XOR, XNOR, and NOT.

### III. RESULTS AND DISCUSSION

#### A. Switching from integer to floating-point mode

The simulation of reconfigurable ALU. In the simulation, there are two types of reconfiguration. The first type switches from integer mode to floating-point mode. The second type switches from floating-point mode to integer mode. Due to the 3-stage pipeline when configured as a floating-point unit and 1-stage pipeline when configured as an integer ALU, the time for switching from integer to floating-point and vice versa is not constant. In the integer mode, the reconfigure input is ‘0’. When the reconfigure input change to ‘1’, the floating-point mode will display the result. If the integer performs add/sub or logic unit operation before switching to the floating-point mode, there are no extra cycles for reconfiguring the reconfigure ALU. However, if the integer mode performs shift operation before switching to the floating-point mode, one cycle is lost for reconfiguration purpose. Since shift operation uses the barrel shifter, which is also used in the first step of the floating-point adder. The simulated result of 64 bit ALU is shown below.

#### B. Switching from floating-point to integer mode

In the floating-point mode, when it switches to the integer mode, which is in shift operation, one reconfiguration cycle is needed. Since a shift operation uses the same barrel shifter, which is also used in the first stage of the floating-point adder. If the add/sub operation done after the floating-point mode, two cycle are needed for switching from the floating-point to the integer mode. However, there is no extra cycle needed for reconfiguring the reconfigurable ALU if switches to logic unit operation. The proposed floating-point adder is based on dual path architecture, and both dynamic and leakage power are reduced by exploiting architecture opportunities to minimize switching activity and maximize the stack effect of the circuits concurrently.

### IV. CONCLUSION

<table>
<thead>
<tr>
<th>TABLE I. POWER AND DELAY OF 64-BIT ALU</th>
</tr>
</thead>
<tbody>
<tr>
<td>POWER CONSUMPTION (MW)</td>
</tr>
<tr>
<td>DELAY (NS)</td>
</tr>
<tr>
<td>NO. OF LUT’S</td>
</tr>
</tbody>
</table>

In this paper, The module was implemented in the Verilog Hardware description language (HDL), tested in simulation using Model Tech’s MODELSIM-ALTRA placed and routed on a Spartan IIe XC2s300e FPGA from Xilinx ISE. When synthesized, this reconfigurable functional unit used 31% number of slices, 5% number of slice flip-flops, and 28% number of 4 input LUTs. For timing report, this module used 20.457 MHz (42.588 ns).

### V. REFERENCES


[3]. Y. Solihin, W. Cameron, L. Yong, D. Lavenier, M. Gokhale, “Mutable Functional Unit and Their Application on


