Implementation of Parallel and Pipeline Scheme in the Standard Floating Point Adder to Improve the Speed

R. Prakash Rao

Abstract: In real time Signal Processing applications, the analogue signal is over sampled as per the Nyquist criterion in order to avoid the aliasing effect. Floating Point (FP) adder is used in the floating point Multiplier Accumulator Content (MAC) for real time Digital Signal Processing (DSP) applications. The heart of any real time DSP processor is floating point MAC. Floating Point MAC is constructed by Finite Impulse Response (FIR) or Infinite Impulse Response (IIR) filters. FIR filters are stable than IIR filters because the impulse response is finite in FIR. Hence, for stable applications FIR filters are preferred. These FIR filters are intern constituted by FP adder, FP multiplier and shifter. In conventional floating point adder the two floating point numbers are added in series. Series means one after the other so the computation speed is less. In series fashion adding the floating point numbers means definitely it furnishes more delay[1] because in the addition of floating point numbers, along with the addition of mantissas; computation is required for both signs and exponents also. Hence, the processing speed is slow for computing the floating point numbers compared with fixed point numbers. Therefore, in order to increase the speed of operation for floating point addition in real time application i.e., to add 16 samples at a time which are in floating notation; a parallel and pipe line technique is going to be incorporated to the two bit floating point architecture. Before developing such novel architecture, a novel algorithm is developed and after, the novel architecture is developed. The total work is simulated by Modelsim 10.3e tool and synthesized by Xilinx 13.6 tool.

Keywords: Over sample, Nyquist criterion, Floating point adder, floating point MAC, parallel and pipe line technique, novel floating point architecture.

I. INTRODUCTION

The enlarge form of MAC is multiplier accumulator content. It is the main core of all the DSP processors. FIR or IIR filters are used to design the MAC. Depending on the user application FIR or IIR filters are used. To design DSP processor, the sub-systems of the MAC are multiplier, adder and shifter. DSP processors are two types i.e., fixed point DSP processors and floating point DSP processors. If the speed is major concern fixed point DSP processors are used where as accuracy is major concern then floating point DSP processors are used. The signal which is to be processed by the DSP processor is to be sampled first as per the Nyquist criterion which says that in order to reconstruct the signal at the output, the input signal is to be oversampled[2] which means that the sampling rate fs should be greater than the twice to that of the fundamental frequency fm. In this work the input signal is sampled for 16 samples. The sampled signals are in floating point. A 16 bit floating point number can be represented by 1 bit of mantissa, 4 bits of exponent and 11 bits of mantissa or fraction. In the general operation of MAC, after receiving the first sample, it will be multiplied by the filter co-efficient and then the result is sent to the adder to add it to the previous value. After addition the output appears as y(n). To get the next sample, a shifting operation is used for the x(n). Like this to add two floating point numbers in the serial fashion an architecture already has been developed in the traditional work. But to add 16 floating point samples at a time, a novel algorithm and its corresponding architecture is certainly required for the DSP applications to increase the speed of operation of floating point processors. Figure1 shows the block diagram of general floating point MAC, in which h(n) indicates filter coefficients.

II. TRADITIONAL FLOATING POINT ADDITION

Figure2 shows the traditional or standard floating point adder. Traditional floating point adder operates in serial fashion[3]. After adding the first two numbers, the result is added to the third sample or third floating point value in the serial manner. Similarly this result is to be added to the fourth sample in the same serial fashion. Like this to add all 16 samples sixteen clock cycles are definitely required by the traditional floating point adder. The traditional floating point adder working principle is well known. It shows the Sx, Sy; Ex, Ey; and Fx, Fy, F indicates the Sign, E indicates the Exponent and F indicates the Fraction or Mantissa. Similarly the suffix x indicates first floating point number and the suffix y indicates second floating point number. Sz is the resultant sign bit, Ez is the resultant exponent and Fz is the resultant fraction or mantissa[4].

Figure1: floating point MAC
Implementation of Parallel and Pipeline Scheme in the Standard Floating Point Adder to Improve the Speed

III. NOVEL ALGORITHM

(Parallel and Pipeline processing of 16-bit floating point Numbers):

Figure 2: Traditional or standard floating point adder

Figure 3 shows transpose and computation of 16 floating point values (each floating point value is having 16-bit length) for parallel and pipeline processing. By transposing the same sample one’s can be added. The 16 floating point values are stored in the 16 x16 matrix. Within the matrix, each floating point value indicates the value of one sample. In order to count the number of 1’s within the same sample, the floating point value is to be transposed. Like this, to count the number of 1’s within the respective sample of all the floating point values, the whole 16x16 matrix has to be transposed. For the example shown in Figure 3, after transpose, the obtained decimal value is (462K)\(_{10}\).

Figure 3: Transpose of all 16-Sample values

Figure 4 shows the parallel and pipeline processing through the counter-cum-adder module. After transpose, the number of 1’s are counted by the counter module through the parallel operation. For example, in Figure 3 after transpose first column consists of 8 one’s, second column consists of 6 one’s third, fourth, fifth columns consists of 7 one’s etc. Its total binary value is 462 k. The same sequence of 8677 7596 9000 0000 are taken in binary (1000 0110 0111 0111 0111 0101 1001 0110 1001 0000 0000 0000 0000 0000 0000 ) and be processed in parallel and pipeline operation and we can observe that the same result of (462K)\(_{10}\).

Figure 4: Parallel and pipeline processing

It is clear that at the end of the parallel operation, the final count value will be passed to the pipeline section. For the first clock pulse, one bit position of the first and alternate numbers are left shifted. The immediate number will be added to the left shifted number. In the similar way during the second clock cycle, 2 bit positions; during the third clock cycle, 4 bit positions and in the fourth clock cycle, 8 bit positions are left shifted. In each level, after left shifted, the immediate number will be added. Figure 4 shows the parallel and pipeline processing for the example of Figure 3. The answer through the parallel and pipeline processing is same as the example shown in Figure 3 which is same as (462K)\(_{10}\). Hence it is proven. The same algorithm is implemented as architecture in Figure 5.

IV. HARDWARE FOR NOVEL ALGORITHM:

The architecture of novel floating point adder requires insert sort module, counter cum adder module and normalize module as shown in below Figure 5.

Figure 5: Architecture of novel floating point adder

Insertion sort module:

While operating with floating point numbers, the operation is to be started with maximum number.
But, after sampling, all the 16-floating point numbers are not in the ascending or descending order. Hence, to get the order of all the 16 samples insertion sort module is used.

**Counter cum Adder module:**

After the multiplication operation, the resultant 16 floating point values are stored in the buffer. By applying insertion sort algorithm, the 16 ascending order samples are applied to the 16 counters. Each sample is having 16 bits hence 4-bit counter($2^4=16$) is used for each sample to count the number of bits. Figure 6. operates as per the algorithm explained before.

**Normalization module:**

Figure 7 explains the normalization operation. In floating point operations at the end to get the correct sign, exponent and mantissa normalization operation is required. After counter cum adder module operation sign bit(Sz) will be generated. The output of the counter cum adder module is applied to the shift register to find the exponent(Ez) and fraction(Fz).

**V. RESULTS AND DISCUSSION**

Figure 8 explains simulation results of traditional or Standard Floating Point Adder. The simulation results for the traditional floating point adder has been performed by the Modelsim 10.3c tool. As the Modelsim tool does not contain floating point packages, we have developed floating point library packages and added with the existed library of the Modelsim tool 10.3c. Hence, by sending the 16-bit floating point values as input, the output is attained.

Figure 9 gives the simulation results of the Novel floating point Adder.

The synthesis has been done by XST tool. The chosen hardware device is Xc2s50e-ft256-6 with speed grade of -6. Table 1 and graph 1 show the results for various parameters of standard floating point adder i.e., addition with serial fashion and novel floating point adder i.e., addition with parallel and pipeline fashion. The input-output pins taken by standard floating point adder and novel floating point adder are 17%. The number of basic elements taken by standard floating point adder are 3.4% and novel floating point adder are 4.16%. The delay of the traditional floating point adder is 0.983ns and the delay of the novel floating point adder is 0.729ns. The speed attained by novel floating point adder is 34.85% more compared with the standard floating point adder which is due to the incorporation of parallel and pipeline architecture. The power consumption for traditional adder is 9.726mW and for novel adder is 11.726mW.
Implementation of Parallel and Pipeline Scheme in the Standard Floating Point Adder to Improve the Speed

Table 1: synthesis report

<table>
<thead>
<tr>
<th>Hardware Resources/Parameters</th>
<th>Standard floating point Adder (addition is in the serial fashion)</th>
<th>Novel floating point Adder(addition is in the parallel and pipeline fashion)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of ICs</td>
<td>31 out of 182 (17%)</td>
<td>31 out of 182 (17%)</td>
</tr>
<tr>
<td>Number of BELs</td>
<td>59 out of 1728 (3.4%)</td>
<td>72 out of 1728 (4.16%)</td>
</tr>
<tr>
<td>Minimum period</td>
<td>0.983 ns</td>
<td>0.729 ns (25.83%)</td>
</tr>
<tr>
<td>Max. Frequency</td>
<td>1017.2 MHz</td>
<td>1371.7 MHz (34.83%)</td>
</tr>
<tr>
<td>Power consumption</td>
<td>9.726 mw</td>
<td>11.726 mw (20.56%)</td>
</tr>
</tbody>
</table>

REFERENCES:

VI. CONCLUSION

In this work both the standard floating point adder and novel floating point adder are synthesized with the Xilinx IST 13.6 software tool. Simulation is done with Modelsim 10.3c software. Modelsim 10.3c tool is not having floating point packages. Hence, a specially developed IEEE 754 standard floating point packages are compiled with the tool and then the synthesis has been performed. The speed of the novel floating point adder is 34.85% more compared with the traditional floating point adder but with the scarifying of some power consumption which is in the order of 20.56%.

AUTHOR’S PROFILE

Dr. R. Prakash Rao, received Ph.D from JNTUH, Hyderabad; M.Tech from College of Engineering Andhra University, Vizag and B.Tech from Nagarjuna University, Guntur, India. He has 14 years of teaching experience. He worked as Professor & HOD in the department of ECE, St.Peter’s Engineering College, Hyderabad. Presently he is working as Associate Professor in Matrusri Engineering College, Saidabad, Hyderabad. He published 18 papers in various National and International Journals and Conferences. He got best faculty award for the A.Y 2017-2018 by the Rotaract club, Secundrabad. He has guided 14 M.Tech projects and about 30 B.Tech projects in various fields. His research interest includes VLSI, VLSI with signal processing, Microwave Engineering and Embedded Systems.