Design and FPGA Implementation of High Performance Rotation Architectures, based on the Radix 4 CORDIC Algorithm

J M Rudagi¹, Basavaraj B Patil¹, Srikant¹ and Subbaraman²

¹Department of Electronics and Communication, KLES' College of Engineering and Technology, Belgaum, Karnataka, India.
²Professor and Dean, Walchand College of Engineering Sangli, Maharashtra, India.

Abstract - CORDIC (Coordinate Rotation Digital Computer) is a well known hardware algorithm for computing various elementary functions. In this work two 32 bit radix 4 CORDIC architectures, unfolded and folded are implemented on available FPGA. The unfolded pipelined architecture consists of a linear array of modules in each of which a micro rotation is carried out. The folded architecture uses pipelining and parallelism. The proposed architectures present a good tradeoff between latency and hardware complexity when compared to already existing CORDIC architectures. The folded architecture requires 25% of the resources as compared to unfolded architecture and operates at a frequency of 56.069MHz hence can be used for real time applications.

Keywords: CORDIC, FPGA, VLSI, Radix 4, pipelined architecture, latency.

1 Introduction

The key concept of CORDIC architecture is based on the simple and the ancient concept of the two dimensional geometry. CORDIC algorithm is originally proposed by Volder [1] in 1959 and later generalized by Walther [2] in 1971 for computation of logarithms, exponentials and square root functions along with trigonometric functions like sine, cosine and tangent. It is an iterative algorithm for the calculation of the rotation of two dimensional vectors in linear, circular and hyperbolic coordinate systems. The current applications of CORDIC algorithm are in the field of DSP, image processing, filtering, matrix algebra etc. The algorithm belongs to the class of linear convergence algorithms and can likewise be implemented using only shift and add operations making it suitable for VLSI implementations.

The number of iterations of conventional radix 2 CORDIC architectures limits it to be used in a high speed application. So the development of high radix CORDIC algorithms is essential for reducing the number of iterations i.e. Reduction of latency with reasonable increment in hardware complexity [6]. The scale factor overhead causes its improvement to be limited. The minimization of the computational...
overhead of scale factor compensation been attempted by different researchers. [7-11].

In this paper two architectures, viz unfolded and folded are implemented on available FPGA. The total number of iterations is reduced by half (n/2). The scale factor compensation is carried out in parallel to the rotation. The resource utilization of the architectures are compared and found that folded architecture requires 25% of resources.

The work is structured as follows. Section 2 describes the radix 4 CORDIC algorithm along with its scale factor compensation. Section 3 describes different architectures, unfolded and folded based on the radix 4 CORDIC algorithm. Section 4, covers implementation details. Section 5 covers comparative study. Concluding remarks are presented in section 6.

2 Radix 4 CORDIC algorithm

The radix 4 CORDIC equations are [4],

\[
\begin{align*}
X_{i+1} &= X_i - \sigma_i 4^i Y_i \\
Y_{i+1} &= Y_i + \sigma_i 4^i X_i \\
Z_{i+1} &= Z_i - \alpha_i[i][\sigma_i]
\end{align*}
\] (1)

Where \(\sigma_i \in \{-2, -1, 0, 1, 2\}\), \(\alpha_i[i][\sigma_i] = \tan^{-1}(\sigma_i r)\), ‘r’ is the radix of the CORDIC. \(X_{i+1}\), \(Y_{i+1}\) are the coordinates of the vector resulting from applying \(i+1\) micro rotation and \(Z_{i+1}\) is the angle to be rotated. The coordinates are scaled by,

\[
K^{-1} = \prod_{i>0} (1+\sigma_i 2^{-2i})^{1/2}
\] (2)

The scale factor depends on \(\sigma_i\). For radix 4 algorithm this value ranges from \(K=1.0\) to \(K=2.52\) [8]. This scale factor must be evaluated for each rotation angle and compensated.

Let us define a variable \(w_i\) as,

\[
w_i = 4^i Z_i
\]

\[A_i[i][\sigma_i] = 4^i \tan^{-1}(\sigma_i * 4^i)
\] (3)

Iteration Z can be expressed as,

\[
W_{i+1} = 4(w_i - A_i[i][\sigma_i])
\] (4)

The equation (4) is required to prove that the variable Z is bounded in each of the iterations. The selection interval for different values of iteration of (1) can be obtained [4].
For $i = 0$,
\[
\sigma_0 = \begin{cases} 
+2 & \text{if } 5/8 \leq \hat{w}_0 < 5/8 \\
+1 & \text{if } 3/8 \leq \hat{w}_0 < 5/8 \\
0 & \text{if } -1/2 \leq \hat{w}_0 < 3/8 \\
-1 & \text{if } -7/8 \leq \hat{w}_0 < -1/2 \\
-2 & \text{if } \hat{w}_0 < -7/8
\end{cases}
\] (5)

For $i > 0$,
\[
\sigma_i = \begin{cases} 
+2 & \text{if } \hat{w}_i \geq 3/2 \\
+1 & \text{if } 1/2 \leq \hat{w}_i < 3/2 \\
0 & \text{if } -1/2 \leq \hat{w}_i < 1/2 \\
-1 & \text{if } -3/2 \leq \hat{w}_i < -1/2 \\
-2 & \text{if } \hat{w}_i < -3/2
\end{cases}
\] (6)

Here $\hat{w}_i$ is the estimate of $w_i$ with three most significant bits. The selection function is true for both carry save redundant arithmetic and non redundant arithmetic [5]

2.1 Scale factor

The final coordinates obtained from the application of the micro rotations (1) do not match the result of rotation. The final coordinates are scaled by a scale factor (2). This factor is not constant as it depends on the sequence of $\sigma_i$'s and has to be calculated for each rotation angle. To provide compensation, the scale factor for each angle has to be calculated and perform direct multiplication to obtain the X and Y coordinates of the rotated vector.

The scale factor can be calculated for $n/4+1$ micro rotation as for rest of the micro rotations it can be taken as one. The scale factors can be stored in a table. The size of the table is $(3^n/4+1* n)$. The Taylor series expansion of (2) is given below,
\[
K^{-1} = 1 - 1/2^i \sigma_i 
\cdot 4^{2i} + 3/8 * \sigma_i 
\cdot 4^{4i} + \ldots \ldots (7)
\]

The $K^{-1}$ can be approximated by the first two terms for $i \geq [n/8+1]$ or three terms if $i \geq [n/12+1]$ Hence we can store only scale factor generated in the first $[n/12+1]$ micro rotations, since the scale factor generated by micro rotations with $i \geq [n/12+1]$ can be calculated by means of addition and shift operations. The size of the table can be reduced to $(3^n/12+1 * n)$. 

3 Radix 4 CORDIC Architectures

In this section we present two 32 bit CORDIC architectures.

3.1 Unfolded Architecture [3]

The pipelined architecture is the efficient method. Since the CORDIC iterations are identical, it is very much convenient to map them into pipelined architectures. The main emphasis in efficient pipelined implementation lies with the minimization of the critical path.

The fig.1 represents the architecture corresponding to a micro rotation that uses the selection function in iteration ‘w’ (fig 2). To enhance throughput, the micro rotations are pipelined into two stages. In order to obtain each coordinate it is necessary to perform an addition and a shift operation.

The scale factor is generated using the method proposed in section 2.1. The scale factor table stores the values of sigma (σ ∈ {0, 1, 2}) and iteration (i ∈ {0 to n/2}). The fig 3 represents unfolded architecture of 32 bit CORDIC processor. The scale factor is calculated in parallel with the micro rotations.
3.2 Folded Architecture: [4]

The architecture is designed for 32 bit precision.

3.2.1 Unscaled CORDIC architecture: This architecture consists of a preprocessing unit for carrying out $\pi/2$ rotations and three processing paths for X, Y and W coordinates as shown in fig 6. It requires sixteen iterations to complete its operation. Every clock cycle produces intermediate values of X and Y.

3.2.2 $K^{-1}$ Calculation: The scale factor (fig 5) is calculated in parallel with X, Y and W coordinates. It requires nine cycles (i.e. $n/4+1$). The generated values of sigma are stored at iterations $i=0$ to $n/4$. These values of sigma with sigma at $i=0$ forming the most significant bits, form an address for ROM (LUT). The LUT stores probable values of $K^{-1}$ for the different combination of sigma $\sigma \in \{0, 1, 2\}$ and iterations $i \in \{0$ to $n/2\}$.

![Fig.4: Scaled architecture](image1)

![Fig.5: $K^{-1}$ Computation](image2)

![Fig.6: Unscaled architecture](image3)
3.2.3 Scaled 32bit CORDIC architecture: Two 32 bit multipliers are used to provide compensated values of X and Y (fig 4). These compensated values will be each of 64 bit word length.

4 Implementation

The CORDIC algorithm is verified using MATLAB tool (version 7.10.0.499, R2010a). Both the architectures have been implemented on the available XC3S400-4PQ208 of the XILINX FPGA (ISE 10.1i). The proposed architectures have been modeled in VERILOG. The combinational path delay of unfolded 32bit architecture is 125.972ns, where as that of folded 32bit architecture is 33.306ns. The latency of the folded 32bit architecture is n/2(i.e. 16) clock cycles. The throughput rate is one valid result per sixteen clock cycles. The folded 32bit architecture operates at 55.906MHz of the clock rate. The resource utilization of FPGA utilization for both architectures is given in Table 1.

The fig. 7 shows the simulation result of scaled CORDIC architecture used for calculating Sine and Cosine values of an angle. It requires 16 cycles to calculate the value.

Fig 7: Simulation Result of scaled 32 bit CORDIC architecture

5 Comparative Study

The architectures are synthesized on different target devices. Table 1 gives comparative study of resource utilization for different target devices. It is observed that the performance of folded architecture is better than unfolded architecture.
Table 1. Comparative study of resource utilization for different target devices

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Unfolded</th>
<th>Folded</th>
<th>Folded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Device</td>
<td>xc3s400-4-pq208</td>
<td>xcv1000e-6bg560</td>
<td>xcv1000e-6bg560</td>
</tr>
<tr>
<td>Number of Slices</td>
<td>2138</td>
<td>4068</td>
<td>541</td>
</tr>
<tr>
<td>Number of Slice flip flops</td>
<td>3983</td>
<td>7497</td>
<td>132</td>
</tr>
<tr>
<td>Number of 4 input LUTs</td>
<td>--</td>
<td>--</td>
<td>1015</td>
</tr>
<tr>
<td>Number of GCLKs</td>
<td>--</td>
<td>--</td>
<td>2</td>
</tr>
<tr>
<td>Number of MULT18X18s</td>
<td>16</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Speed-MHz</td>
<td>--</td>
<td>--</td>
<td>55.906</td>
</tr>
<tr>
<td>Combinational path delay -ns</td>
<td>125.972</td>
<td>181.912</td>
<td>33.306</td>
</tr>
<tr>
<td>Bit precision-bits</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
</tbody>
</table>

6 Conclusion

Two 32bit CORDIC architectures have been implemented on the available XILINX FPGA. The folded architecture operates at a frequency of 55.906MHz. Latency of the architecture is n/2 (i.e.16) clock cycles. The folded architecture requires 25% of the resources as compared to unfolded architecture. The folded architecture with optimized speed and area can be used for real time applications.

References