**CE4518 – Analysis and Simulation of a Rotation-Mode CORDIC Processor**

**A picture containing refrigerator

Description automatically generated**

**Group Members:** Brian Bulfin (16157516), Rory Loughrey (16178041), Micheal O’Connor (16188713)

**Background**

Enter here

**Implementation**

Enter here

**Results**

A picture containing screen, monitor, clock

Description automatically generated

***Figure 1:*** *Waveform representations of most important CORDIC signals. Angles are represented in hexadecimal.*

Figure 1 illustrates the operation of the CORDIC Verilog module and displays the functionality of the INIT and DONE signals. The performance of our implementation was assessed by analysing its accuracy. First, the C program was used to find the sine and cosine of every possible angle from -π/2 to π/2, which was possible due to the 2.16 fixed point representation used. Each output was then compared to the outputs generated by the standard C math sin and cos functions to find the maximum errors and their associated angles. The program was run for 17 iterations at first, as this is the maximum possible given the 2.16 representation since Δa17 = 2-16, the smallest allowed by 2.16 fixed point.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Cos** | **Sin** | **Combined** |
| **Error** | -0.000173 | -0.000168 | 0.000263 |
| **Angle** | 1.555954 | 0.014847 | -0.014847 |

***Table 1:*** *Max errors and associated angles of the CORDIC C implementation for i < 17.*

Table 1 shows that the maximum errors for both sine and cosine occur near angles of 0 and π/2 respectively, where the outputs of the functions approach 0. At this point, the number of iterations was reduced until the max error was minimised, which occurred at 8 iterations.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Cos** | **Sin** | **Combined** |
| **Error** | 0.000073 | 0.000068 | 0.000131 |
| **Angle** | 1.486771 | 0.084030 | 1.486771 |

***Table 2:*** *Max errors and associated angles of the CORDIC C implementation for i < 8.*

The results presented in table 2 show that the max error dropped significantly by reducing the number of iterations. Once these errors and angles were calculated, the accuracy of the Verilog implementation could be found.

|  |  |  |  |
| --- | --- | --- | --- |
| **Cos(1.486771)** | | **Sin(0.08403)** | |
| **Actual** | 0000010101**0**1111100 | **Actual** | 0000010101**0**1111101 |
| **CORDIC** | 0000010101**1**0000001 | **CORDIC** | 0000010101**1**0000001 |

***Table 3:*** *Binary representations of worst-case cos and sin angles at 8 CORDIC iterations.*

Table 3 demonstrates the implementation is reliably accurate to at least 10 bits, or 8 decimal places. Due to the fixed number of iterations the results can never be guaranteed to be accurate. Even if the correct angle, sine, and cosine are found after a certain number of iterations the algorithm will persist and sacrifice accuracy as a result.

**Conclusions**

The speed, efficiency, and simplicity of implementation make the rotation-mode CORDIC algorithm a good choice for low-complexity systems such as microcontrollers and FPGA’s without expensive hardware multipliers. However, the fundamental accuracy limitation enforced by the fixed number of iterations means that this algorithm should not be used for high-precision applications, especially as the previously mentioned limitations become less relevant as hardware continues to become cheaper.