I was tasked with building and implementing a Cordic math system. The Cordic equations that I was given to implement were:

Where

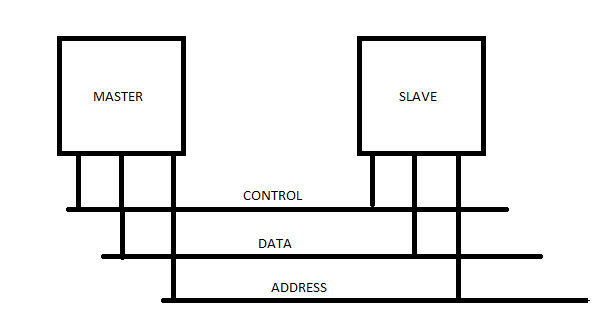
Using these equations I created a register to do the calculations. Depending on the leading sign bit of “Z” the register would do the necessary calculations and shifting. The alpha values were hardcoded into the VHDL, this was done because the VHDL language actually uses the Cordic math system to calculate trigonometric values such as arctan. It made no sense to me to make a Cordic register to be used in a Cordic system that calculates Cordic values.

I was able to write a simple C program that calculated the arctan values for alpha and printed them all out, rather than imputing them one by one.

The next step was to make a state table to show what operation had to be completed to move to the next state, and which state would do what operation. The diagram below illustrates how my state machine works and when states are switched.

Included are my “CordicResults.txt” file and one complete cycle of the wave form of my project.

A simple block diagram of how the system is implemented



The implementation is fairly simple, when the system begins, nothing is happening. When the master asserts request and a write address, the slave acknowledges it and then begins the computations. As soon as it has finished, the final values are stored in vectors, which are remembered until the master asserts a request for the data and an address for it. Once this happens the slave acknowledges this and then provides the data to the requested address.   
This happens 3 times, once for X, Y, and Z values.