CORDIC Lab

Design

Design and implement a flexible CORDIC generator in Chisel.

1) Support vectoring and rotation modes
2) Make widths for `X`, `Y`, and `Z` parameterized
3) Gain correction will be optionally supported (controlled by a parameter). If gain correction is enabled, the output will be multiplied by the appropriate scaling factor. Otherwise, the unscaled result will be passed directly to the output.
4) `Z` is modulo 2π with full range supported
5) The degree of unrolling should be set by a parameter.

Template

Template code is in your repo. Read through the template and get a sense of what's there.

Parameters and IO

`Cordic.scala` defines some parameters objects, a `Bundle` to use as your IO, and an empty implementation of a FixedIterativeCordic (which you will need to implement).

CordicApp

The template also includes an `App` (similar to a main function in Java) that can be used to run your generator. It accepts commandline arguments. To see the options, run

```
sbt "run --help"
```

Tests

The template includes a simple test using `DspTester` and an associated ScalaTest specification. You can run tests with

```
sbt test
```

Deliverables

1) ***CORDIC implementation in Chisel:*** Your generator should be parameterized as described in the design section.
2) ***Tests:*** A test has been written, but it doesn't test that much. Write more tests to check that every mode works for a variety of parameterizations.
3) ***Documentation:*** Add a block diagram representing your design and a description to your README.

Make sure to push all your changes and deliverables to your private repo.