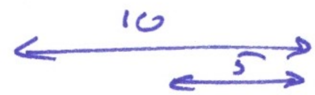
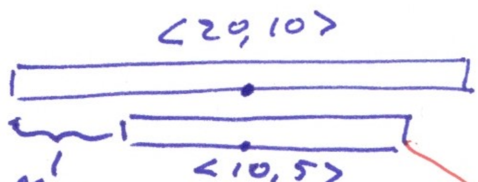
02/19/19 12:28:19 C:\Users\Patrick Schaumont\Desktop\handout-9

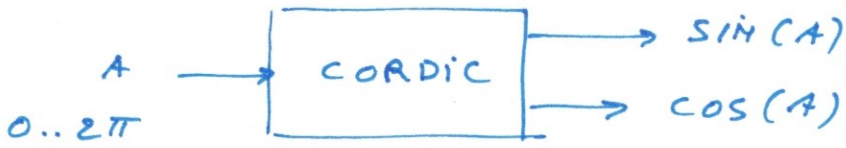
lecture09 - Fixed Point Computations in Verilog

```
// Write a module that computes the
// quadratic expression
//
//    f(x) = 0.85 * x^2 + 1
//
// The input and output data is a signed fix<10, 5>
// In case of overflow, the module returns 0
//
// You have to complete the dots with Verilog expressions

module quadratic(input SIGNED [9:0]   in,

                 output SIGNED [9:0] out);

  parameter CONST_0_85 = (0.85 * 32);

  parameter CONST_1    = (1    * 32);

  wire SIGNED [19:0] square;
  wire   ovfsquare, ovfsign;

  assign square    = in * in;
                                                all 0
                                              or all 1
  // ovfsquare is high when an overflow occured

  assign ovfsquare = (SQUARE[19:15] != 5'b11111) ||
                     (SQUARE[19:15] != 5'b00000);
                     CATCH SIGN OVF:
                       SQUARE[19:14] != 6'b111111
  wire signed [19: 0] cmul;

  assign cmul = CONST_0_85 * square[14:5];

  wire signed [9:0] result;

  assign result = cmul[14:5] + CONST_1;

  assign ovfsign    = result[9];

  assign out = (ovfsquare || ovfsign) ? 10'd0 : result;

endmodule
```





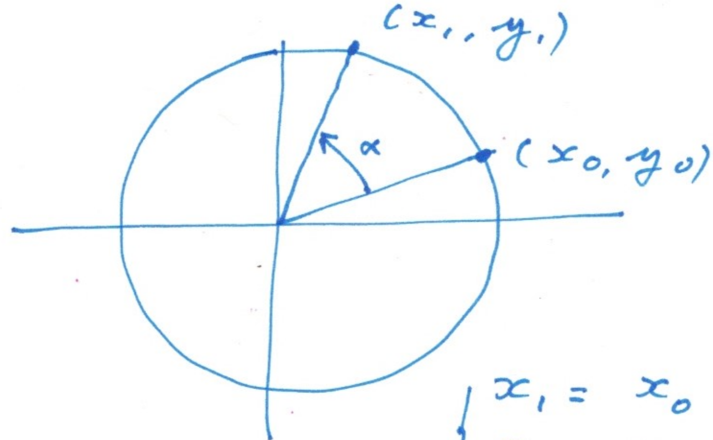
CORDIC



IDEA OF CORDIC:

APPROXIMATE A BY A SEQUENCE OF SMALLER ANGLES

ROTATION:



$$\begin{cases} x_1 = x_0 \cos\alpha - y_0 \sin\alpha \\ y_1 = x_0 \sin\alpha + y_0 \cos\alpha \end{cases}$$

$$\begin{cases} x_1 = \frac{1}{\cos\alpha}\left(x_0 - y_0 \tan\alpha\right) \\ y_1 = \frac{1}{\cos\alpha}\left(x_0 \tan\alpha + y_0\right) \end{cases}$$

CHOOSE $\alpha = \alpha_0 \pm \alpha_1 \pm \alpha_2 \pm \alpha_3 \ldots$

α_i SUCH THAT $\tan(\alpha_i) = \frac{1}{2^i}$

SUCESSIVE APPROXIMATION OF α