

SINE WAVE GENERATOR

Md. Aamir Sohail, Piyush Udhan

EE5811: FPGA LAB
&
EE3025: Independent Project

Department of Electrical Engineering,
IIT HYDERABAD

April 25, 2019

Project Outline

The aim of our project is to make a sine wave generator using Icoboard FPGA. We have used cordic algorithm to do so along with arduino Mega for post processing. To realise the sine wave, we have used an R2R DAC circuit to convert digital signal to analog and a digital oscilloscope (Analog Discovery) to see the waveform on the screen.

- All the verilog and arduino codes, the report and presentation are available on the github link which is a free open-source. [► Github Link](#)

Software setup

- ➊ Open the command terminal execute the following command [▶ Link](#) for installing: [1]
 - wiringPi
 - IcoProg
 - IcoStorm tools
 - Arachne-pnr
 - Yosys
- ➋ Creating **Makefile** [1]

MOTIVATION

If a unit vector with co-ordinates $(x_1, y_1) = (1, 0)$ is rotated by an angle θ , its new co-ordinate will be $(x_2, y_2) = (\cos \theta, \sin \theta)$. Thus, by finding the (x_2, y_2) , $\cos \theta, \sin \theta$ can easily be computed.

Circular CORDIC Algorithm

CORDIC iteration: [2]

$$x(i+1) = x(i) - d_i y(i) 2^{-i} \quad (1)$$

$$y(i+1) = y(i) + d_i x(i) 2^{-i} \quad (2)$$

$$\alpha(i+1) = \alpha(i) - d_i \tan^{-1} 2^{-i} \quad (3)$$

Choose $d_i \in \{+1, -1\}$ such that $\alpha(n) \rightarrow 0$

After n pseudorotation steps, when $\alpha(n)$ is well enough close to zero, we will get $\sum \theta(i) = \alpha$. Finally the CORDIC iterations in ROTATION MODE become:

$$x(n) = \mathbf{K} (x \cos \alpha - y \sin \alpha) \quad (4)$$

$$y(n) = \mathbf{K} (y \cos \alpha + x \sin \alpha) \quad (5)$$

$$\alpha(n) = 0 \quad (6)$$

where $\mathbf{K} = \prod (1 + \tan^2 \theta(i))^{1/2}$, $\theta(i) = \tan^{-1}(d_i 2^{-i})$

Domain of Convergence

- In rotation mode, convergence of $\alpha(n)$ to 0 is possible because each CORDIC angle is more than half the previous angle.

$$\tan^{-1}(2^{-(i+1)}) \geq 0.5 * \tan^{-1}(2^{-i})$$

- Thus, domain of convergence is $-99.7^\circ \leq \alpha \leq 99.7^\circ$ where $\pm 99.7^\circ$ is the sum of the CORDIC angle.
- Beyond the above range, we can use Trigonometric identities to convert the problem within the domain of convergence.

Implementation of Circular CORDIC in RTL

- The digital design should be able to compute Sine and Cosine of the input angles $\in [0, 2\pi]$ including the floating point angles like 89.45° , 29.7° etc.
- A 16-bit binary scaling system has been used to represent angles. The resolution of the design is $360^\circ / 2^{16} = 5.49316406 \times 10^{-3}$. Angle $^\circ$ to 16-bit binary value conversion:

$$(16bit)\text{angle} = \frac{2^{16} * \text{angle}^\circ}{360^\circ}$$

- The upper two bits represent the quadrant [3].
 - 2b'00 -> represents I quadrant i.e $(0 - \pi/2)$ range
 - 2b'01 -> represents II quadrant i.e $(\pi/2 - \pi)$ range
 - 2b'10 -> represents III quadrant i.e $(\pi - 3\pi/2)$ range
 - 2b'11 -> represents IV quadrant i.e $(3\pi/2 - 2\pi)$ range

Coding and Hardware Environment

- CORDIC Algorithm: Verilog
 - ① Xilinx ISE-Design Suite 14.7¹
 - ② Simulation: Xilinx-ISim software
- Postprocessing:
 - ① Software: Arduino [▶ Download Link](#)
 - ② Board: Arduino MEGA 2560
- Oscilloscope:
 - ① DAC: 'R2R Ladder' [▶ Link](#)
 - ② Analog Discovery (Digital Oscilloscope)²
 - Software: Waveform [▶ Download Link](#)

¹L.Gregg - How to Download & Install Xilinx ISE 14.7 Windows 10 [▶ Youtube Link](#)

²Getting Started with Analog Discovery [▶ Youtube Link](#)

Circuit and System Setup

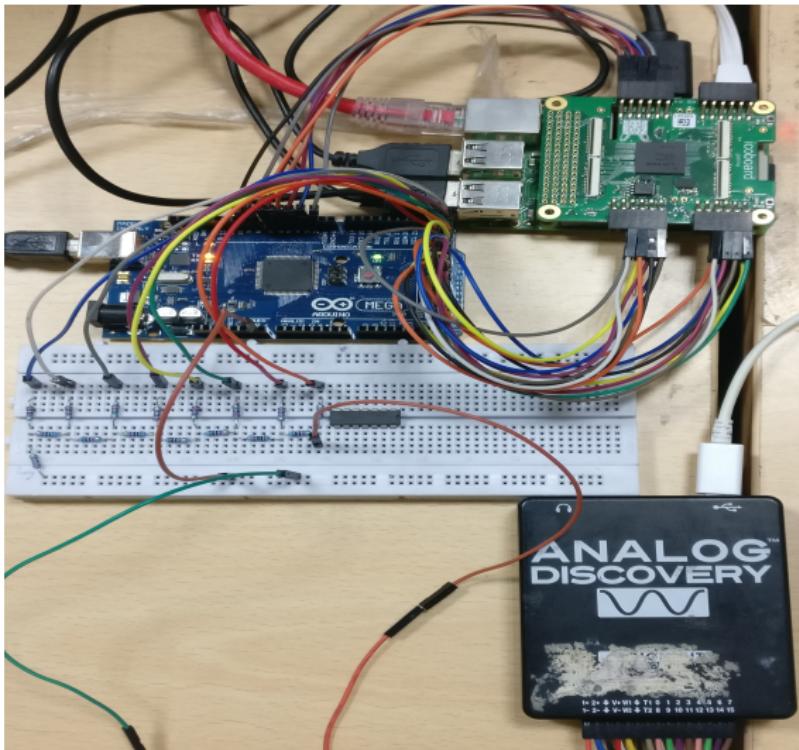


Figure: Circuit Arrangement

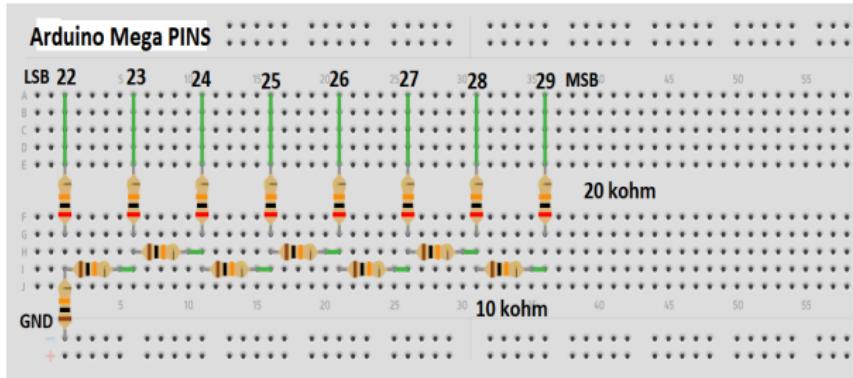


Figure: 8-bit 'R2R' DAC Circuit

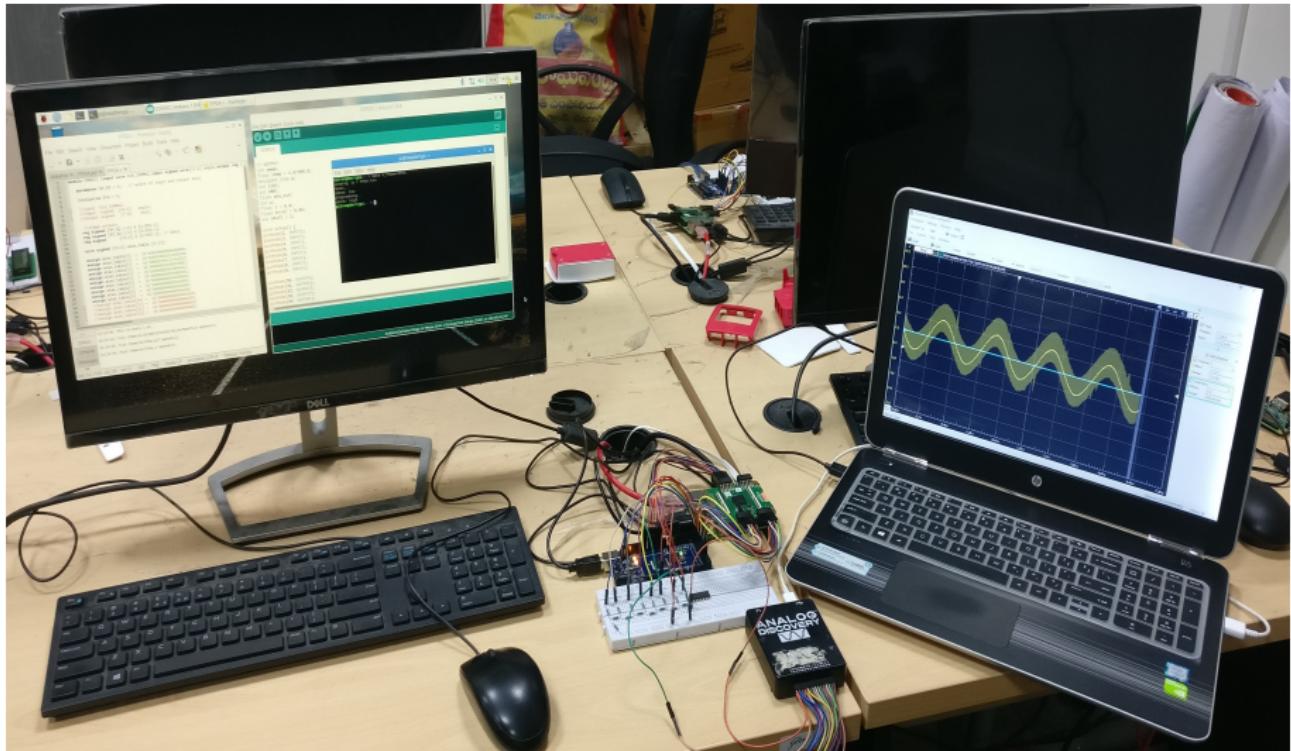


Figure: System Setup

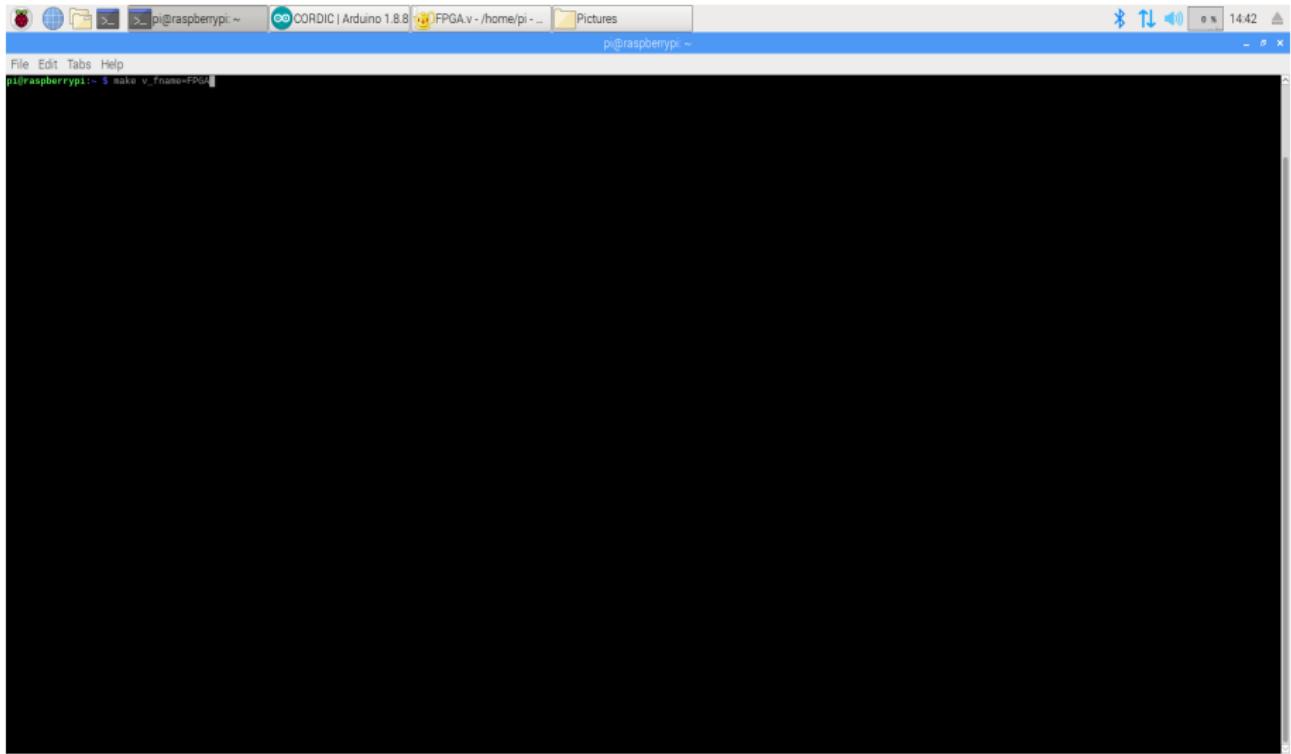


Figure: Uploading FPGA.v to Ico-Board FPGA

```
pi@raspberrypi: ~
```

```
File Edit Tabs Help
```

```
Number of public wires: 3
Number of public wire bits: 25
Number of memories: 0
Number of memory bits: 0
Number of processes: 0
Number of cells: 869
    SB_CARRY 313
    SB_DFF 8
    SB_LUT4 539

2.28. Executing CHECK pass (checking for obvious problems).
Checking module CORDIC..
found and reported 0 problems.

2.29. Executing BLIF backend.

Warnings: 3 unique messages, 3 total
End of script. Logfile hash: fed7794a91
CPU: user 21.40s system 0.41s, MEM: 21.98 MB total, 16.12 MB resident
Yosys 0.8+119 (git sha1 ffc6e2bf, clang 3.8.1+24+rp1 -fPIC -Os)
Time spent: 20s 30x opt_clean (4 sec), 16s 30x opt_expr (3 sec), ...
arachne-pnr -d 8k -p FPGApcf -o FPGA.asc FPGA.blif
seed: 1
device: 8k
read_chipdb +/share/arachne-pnr/chipdb-8k.bin...
    supported packages: bgl21, bgl21:4k, cb132, cb132:4k, cm121, cm121:4k, cm225, cm225:4k, cm@1, cm@1:4k, ct256, tq144:4k
    read_blif FPGA.blif...
    promote...
    read_pcf FPGApcf...
    instantiate_io...
    pack...

After packing:
IOs      25 / 206
G0s      0 / 0
G0..IOs   0 / 0
LCs      677 / 7680
DFF     325
CARRY, DFF 0
DFF PASS 0
CARRY PASS 65
BRAMs    0 / 32
NHRMBOOTs 0 / 1
PLLs     0 / 2

place_constraints...
promote_globals...
    promoted CLK_100MHZS2, 0 / 0
    promoted 1 nets
        1 clk
    1 globals
        1 clk
realize_constants...
    realized 1
place...
    initial wire length = 22631
    at iteration #50: temp = 28.0339, wire length = 12001
```

Figure: Uploading FPGA.v to Ico-Board FPGA

The screenshot shows a terminal window titled 'pi@raspberrypi:~\$' running a screen session. The command 'screen' is being typed at the prompt.

```

pi@raspberrypi:~$ screen

```

Below the terminal window, the status bar indicates:

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on /dev/ttyACM0

The main window contains two code editors. The left editor is titled 'FPGA.v - /home/pi - Geany' and shows Verilog code for aCORDIC module. The right editor is titled 'CORDIC | Arduino 1.8.8' and shows C++ code for aCORDIC module. Both editors have tabs for 'File', 'Edit', 'Sketch', 'Tools', and 'Help'.

```

// module CORDIC (input wire CLK_100MHz, input signed wire[15:0] angle, output reg [15:0] swap);
parameter XY_SZ = 8; // width of input and output data
localparam STS = 8;
//stage outputs
reg signed [XY_SZ-1:0] X [0:STS-1];
reg signed [XY_SZ-1:0] Y [0:STS-1];
reg signed [15:0] Z [0:STS-1]; // 32bit
wire signed [15:0] atan_table [0:13];
assign atan_table[0] = 16'b0010000000000000;
assign atan_table[1] = 16'b0001001011110010;
assign atan_table[2] = 16'b0000100111110011;
assign atan_table[3] = 16'b0000010100001001;
assign atan_table[4] = 16'b0000001000000011;
assign atan_table[5] = 16'b0000000100000010;
assign atan_table[6] = 16'b0000000010000010;
assign atan_table[7] = 16'b0000000001000010;
assign atan_table[8] = 16'b0000000000100000;
assign atan_table[9] = 16'b0000000000010000;
assign atan_table[10] = 16'b0000000000001000;
assign atan_table[11] = 16'b0000000000000101;
assign atan_table[12] = 16'b0000000000000010;
assign atan_table[13] = 16'b0000000000000001;

always @ (posedge CLK_100MHz) // 199981.6467 = 32766: 32766 but extra one bit
begin
    //if (1 < 0)
    //begin
        case (angle[15:14])
            2'b00:
            2'b11: // no pre-rotation needed for these quadrants
            begin
                // X[n], Y[n] is 1 bit larger than Xin, Yin, but Verilog handles it
                $display("angle = %d ",angle);
            end
        endcase
    end
endmodule

```

```

File Edit Sketch Tools Help
CORDIC
File Edit Tabs Help
place time 26.49s
route...
pass 1, 0 shared.
After routing:
span_4 667 / 26096
span_32 163 / 5632
route time 15.47s
write.txt FPGA.asc
pinNode(2, INPUT);
pinNode(3, INPUT);
pinNode(4, INPUT);
pinNode(5, INPUT);
pinNode(6, INPUT);
pinNode(7, INPUT);
pinNode(8, INPUT);
pinNode(9, INPUT);
pinNode(30, OUTPUT);
pinNode(31, OUTPUT);
pinNode(32, OUTPUT);
pinNode(33, OUTPUT);
pinNode(34, OUTPUT);
pinNode(35, OUTPUT);
pinNode(36, OUTPUT);
pinNode(37, OUTPUT);
pinNode(38, OUTPUT);
pinNode(39, OUTPUT);
pinNode(40, OUTPUT);

void setup() {
    pinMode(2, INPUT);
    pinMode(3, INPUT);
    pinMode(4, INPUT);
    pinMode(5, INPUT);
    pinMode(6, INPUT);
    pinMode(7, INPUT);
    pinMode(8, INPUT);
    pinMode(9, INPUT);
    pinMode(30, OUTPUT);
    pinMode(31, OUTPUT);
    pinMode(32, OUTPUT);
    pinMode(33, OUTPUT);
    pinMode(34, OUTPUT);
    pinMode(35, OUTPUT);
    pinMode(36, OUTPUT);
    pinMode(37, OUTPUT);
    pinMode(38, OUTPUT);
    pinMode(39, OUTPUT);
    pinMode(40, OUTPUT);
}

```

Figure: Uploading FPGA.v to Ico-Board FPGA

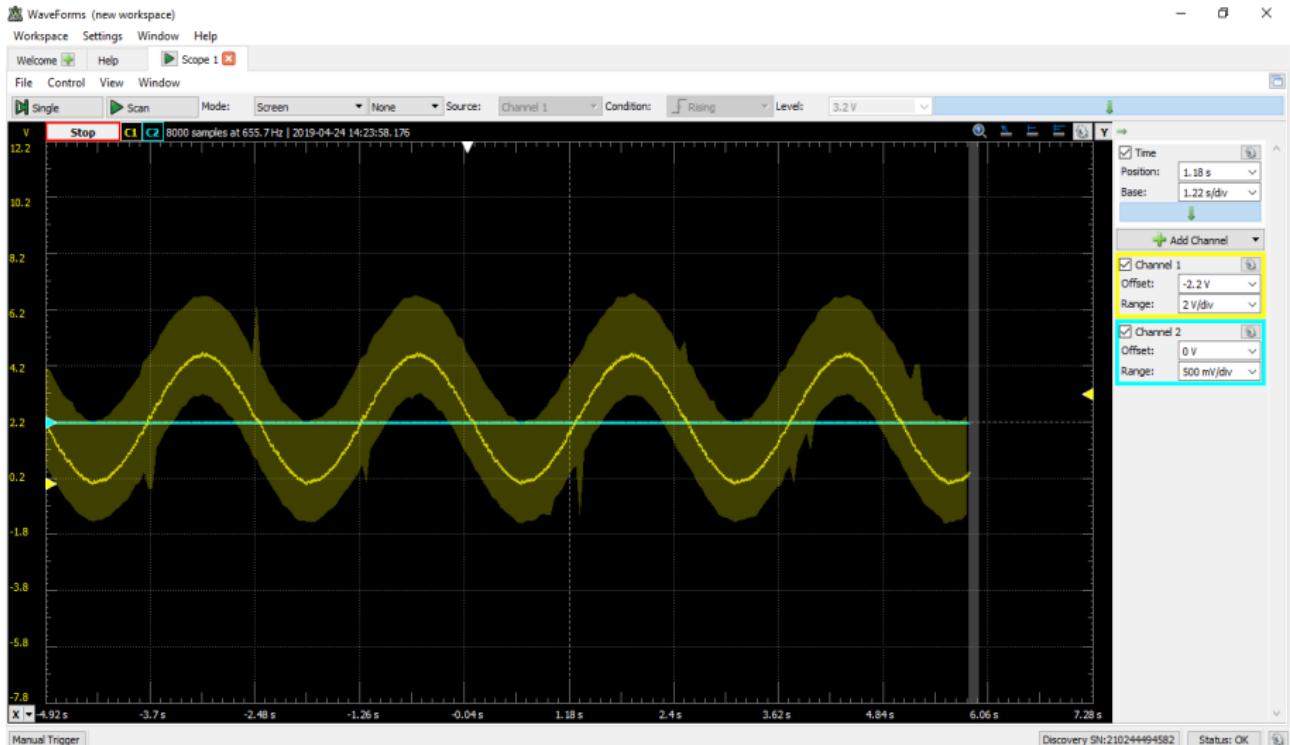


Figure: Output Waveform

References



GVV Sharma, Hemanth Kumar Desinnedi,

"Introduction to verilog programming on IcoBoard FPGA", [▶ Link](#)



Behrooz.Parhami., COMPUTER ARITHMETIC-Algorithms and Hardware Designs
2nd edition., OXFORD UNIVERSITY PRESS, 2010, ch.22.



Kirk Weedman, CORDIC Design and Simulation using Verilog, [▶ Link](#)