End Semester Project Report

By Swadhin Agrawal (Roll. No. 15223)

Digital circuits and systems

1. Overview

This report explains the design and simulation of a sine wave generator which can generate a sine wave of frequency equal to 230Hz using a clock of 1MHz. This report includes the Verilog code for the simulation along with simulation results and output waveforms. In the end I have calculated the purity of the generated sine signal in terms of total harmonic distortion. I have done this in two different approaches, one where I used the available codes in the internet (Method 1) by modifying it to get output of 230Hz using clock of 1MHz and another approach where I coded myself (Method 2) to get the desired output. I did this in two different approaches to see how the purity differs in both different ways. In the end I have described the circuit design of my device.

2. Verilog Code with explanations

- Method 1:

```
1   `timescale 100ns/100ns
2   module testbench();
3
4     reg clk_1mhz, reset;
5
6     reg [31:0] index;
7     wire signed [15:0]  testbench_out;
8
9     //Initialize clocks and index
10    initial begin
11            clk_1mhz = 1'b0;
12            index  = 32'd0;
13            //testbench_out = 15'd0 ;
14    end
15
16    //Toggle the clocks
17    always begin
18            #5
19            clk_1mhz  = !clk_1mhz;
20    end
21
22    //Initialize and drive signals
23    initial begin
24            reset  = 1'b0;
25            #100
26            reset  = 1'b1;
27            #300
28            reset  = 1'b0;
29    end
30
31    //Increment index
32    always @ (posedge clk_1mhz) begin
33            index  <= index + 32'd1;
34    end
35
```

```
36   //Instantiation of Device Under Test
37   // hook up the sine wave generators
38 DDS DUT    (.clock(clk_1mhz),
39          .reset(reset),
40                              .increment({18'h003C, 14'b0}),
41                              .phase(8'd0),
42                              .sine_out(testbench_out));
43
44 endmodule
45
46 ////////////////////////////////////////////////
47 ////////////     Direct Digital Synth   //////////////
48 ////////////////////////////////////////////////
49 // Input is an increment, phase and a clock
50 // output is a sine wave
51 // Output frequency = increment * clock_rate /
   accumulator_bit_length
52 // Here accumulator_bit_length is 32 bits
53 // Phase is measured in samples out of 256/cycle. e.g. 64
   input is 90 degrees
54 module DDS (clock, reset, increment, phase, sine_out);
55 input clock, reset;
56 input [31:0] increment ;
57 input [7:0]  phase;
58 output wire signed [15:0] sine_out;
59 reg [31:0] accumulator;
60
61 always@(posedge clock) begin
62   if (reset) accumulator <= 0;
63   // increment phase accumulator
64   else accumulator <= accumulator + increment  ;
65 end
66
67 // link the accumulator to the sine lookup table
68 sync_rom sineTable(clock, accumulator[31:24]+phase,
   sine_out);
69
70 endmodule
71 ////////////////////////////////////////////////
72 ////////////     Sin Wave ROM Table     //////////////
73 ////////////////////////////////////////////////
74 // produces a 2's comp, 16-bit, approximation
75 // of a sine wave, given an input phase (address)
76 module sync_rom (clock, address, sine);
77 input clock;
78 input [7:0] address;
79 output [15:0] sine;
80 reg signed [15:0] sine;
81 always@(posedge clock)
82 begin
83     case(address)
84                   8'h00: sine = 16'h0000 ;
85                   8'h01: sine = 16'h0192 ;
86                   8'h02: sine = 16'h0323 ;
87                   8'h03: sine = 16'h04b5 ;
88                   8'h04: sine = 16'h0645 ;
89                   8'h05: sine = 16'h07d5 ;
90                   8'h06: sine = 16'h0963 ;
91                   8'h07: sine = 16'h0af0 ;
92                   8'h08: sine = 16'h0c7c ;
93                   8'h09: sine = 16'h0e05 ;
```

```
94                  8'h0a: sine = 16'h0f8c ;
95                  8'h0b: sine = 16'h1111 ;
96                  8'h0c: sine = 16'h1293 ;
97                  8'h0d: sine = 16'h1413 ;
98                  8'h0e: sine = 16'h158f ;
99                  8'h0f: sine = 16'h1708 ;
100                        8'h10: sine = 16'h187d ;
101                        8'h11: sine = 16'h19ef ;
102                        8'h12: sine = 16'h1b5c ;
103                        8'h13: sine = 16'h1cc5 ;
104                        8'h14: sine = 16'h1e2a ;
105                        8'h15: sine = 16'h1f8b ;
106                        8'h16: sine = 16'h20e6 ;
107                        8'h17: sine = 16'h223c ;
108                        8'h18: sine = 16'h238d ;
109                        8'h19: sine = 16'h24d9 ;
110                        8'h1a: sine = 16'h261f ;
111                        8'h1b: sine = 16'h275f ;
112                        8'h1c: sine = 16'h2899 ;
113                        8'h1d: sine = 16'h29cc ;
114                        8'h1e: sine = 16'h2afa ;
115                        8'h1f: sine = 16'h2c20 ;
116                        8'h20: sine = 16'h2d40 ;
117                        8'h21: sine = 16'h2e59 ;
118                        8'h22: sine = 16'h2f6b ;
119                        8'h23: sine = 16'h3075 ;
120                        8'h24: sine = 16'h3178 ;
121                        8'h25: sine = 16'h3273 ;
122                        8'h26: sine = 16'h3366 ;
123                        8'h27: sine = 16'h3452 ;
124                        8'h28: sine = 16'h3535 ;
125                        8'h29: sine = 16'h3611 ;
126                        8'h2a: sine = 16'h36e4 ;
127                        8'h2b: sine = 16'h37ae ;
128                        8'h2c: sine = 16'h3870 ;
129                        8'h2d: sine = 16'h3929 ;
130                        8'h2e: sine = 16'h39da ;
131                        8'h2f: sine = 16'h3a81 ;
132                        8'h30: sine = 16'h3b1f ;
133                        8'h31: sine = 16'h3bb5 ;
134                        8'h32: sine = 16'h3c41 ;
135                        8'h33: sine = 16'h3cc4 ;
136                        8'h34: sine = 16'h3d3d ;
137                        8'h35: sine = 16'h3dad ;
138                        8'h36: sine = 16'h3e14 ;
139                        8'h37: sine = 16'h3e70 ;
140                        8'h38: sine = 16'h3ec4 ;
141                        8'h39: sine = 16'h3f0d ;
142                        8'h3a: sine = 16'h3f4d ;
143                        8'h3b: sine = 16'h3f83 ;
144                        8'h3c: sine = 16'h3fb0 ;
145                        8'h3d: sine = 16'h3fd2 ;
146                        8'h3e: sine = 16'h3feb ;
147                        8'h3f: sine = 16'h3ffa ;
148                        8'h40: sine = 16'h3fff ;
149                        8'h41: sine = 16'h3ffa ;
150                        8'h42: sine = 16'h3feb ;
151                        8'h43: sine = 16'h3fd2 ;
152                        8'h44: sine = 16'h3fb0 ;
153                        8'h45: sine = 16'h3f83 ;
154                        8'h46: sine = 16'h3f4d ;
```

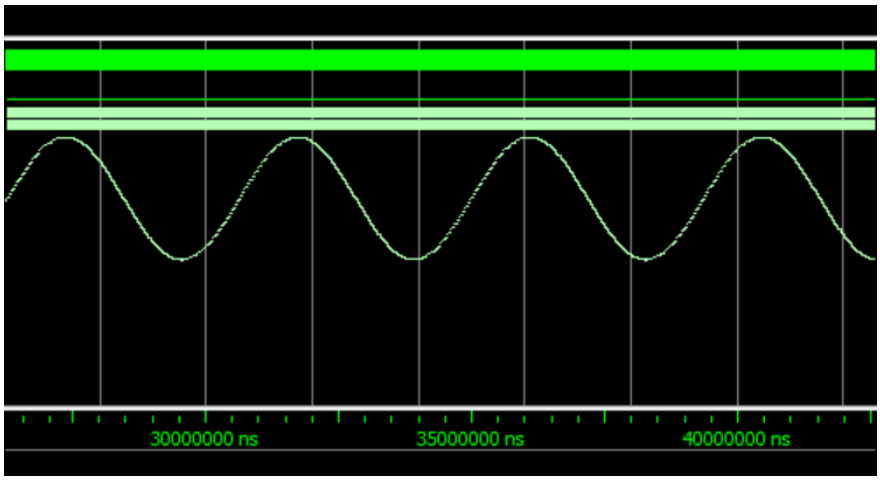
```
155                         8'h47: sine = 16'h3f0d ;
156                         8'h48: sine = 16'h3ec4 ;
157                         8'h49: sine = 16'h3e70 ;
158                         8'h4a: sine = 16'h3e14 ;
159                         8'h4b: sine = 16'h3dad ;
160                         8'h4c: sine = 16'h3d3d ;
161                         8'h4d: sine = 16'h3cc4 ;
162                         8'h4e: sine = 16'h3c41 ;
163                         8'h4f: sine = 16'h3bb5 ;
164                         8'h50: sine = 16'h3b1f ;
165                         8'h51: sine = 16'h3a81 ;
166                         8'h52: sine = 16'h39da ;
167                         8'h53: sine = 16'h3929 ;
168                         8'h54: sine = 16'h3870 ;
169                         8'h55: sine = 16'h37ae ;
170                         8'h56: sine = 16'h36e4 ;
171                         8'h57: sine = 16'h3611 ;
172                         8'h58: sine = 16'h3535 ;
173                         8'h59: sine = 16'h3452 ;
174                         8'h5a: sine = 16'h3366 ;
175                         8'h5b: sine = 16'h3273 ;
176                         8'h5c: sine = 16'h3178 ;
177                         8'h5d: sine = 16'h3075 ;
178                         8'h5e: sine = 16'h2f6b ;
179                         8'h5f: sine = 16'h2e59 ;
180                         8'h60: sine = 16'h2d40 ;
181                         8'h61: sine = 16'h2c20 ;
182                         8'h62: sine = 16'h2afa ;
183                         8'h63: sine = 16'h29cc ;
184                         8'h64: sine = 16'h2899 ;
185                         8'h65: sine = 16'h275f ;
186                         8'h66: sine = 16'h261f ;
187                         8'h67: sine = 16'h24d9 ;
188                         8'h68: sine = 16'h238d ;
189                         8'h69: sine = 16'h223c ;
190                         8'h6a: sine = 16'h20e6 ;
191                         8'h6b: sine = 16'h1f8b ;
192                         8'h6c: sine = 16'h1e2a ;
193                         8'h6d: sine = 16'h1cc5 ;
194                         8'h6e: sine = 16'h1b5c ;
195                         8'h6f: sine = 16'h19ef ;
196                         8'h70: sine = 16'h187d ;
197                         8'h71: sine = 16'h1708 ;
198                         8'h72: sine = 16'h158f ;
199                         8'h73: sine = 16'h1413 ;
200                         8'h74: sine = 16'h1293 ;
201                         8'h75: sine = 16'h1111 ;
202                         8'h76: sine = 16'h0f8c ;
203                         8'h77: sine = 16'h0e05 ;
204                         8'h78: sine = 16'h0c7c ;
205                         8'h79: sine = 16'h0af0 ;
206                         8'h7a: sine = 16'h0963 ;
207                         8'h7b: sine = 16'h07d5 ;
208                         8'h7c: sine = 16'h0645 ;
209                         8'h7d: sine = 16'h04b5 ;
210                         8'h7e: sine = 16'h0323 ;
211                         8'h7f: sine = 16'h0192 ;
212                         8'h80: sine = 16'h0000 ;
213                         8'h81: sine = 16'hfe6e ;
214                         8'h82: sine = 16'hfcdd ;
215                         8'h83: sine = 16'hfb4b ;
```

```
216                         8'h84: sine = 16'hf9bb ;
217                         8'h85: sine = 16'hf82b ;
218                         8'h86: sine = 16'hf69d ;
219                         8'h87: sine = 16'hf510 ;
220                         8'h88: sine = 16'hf384 ;
221                         8'h89: sine = 16'hf1fb ;
222                         8'h8a: sine = 16'hf074 ;
223                         8'h8b: sine = 16'heeef ;
224                         8'h8c: sine = 16'hed6d ;
225                         8'h8d: sine = 16'hebed ;
226                         8'h8e: sine = 16'hea71 ;
227                         8'h8f: sine = 16'he8f8 ;
228                         8'h90: sine = 16'he783 ;
229                         8'h91: sine = 16'he611 ;
230                         8'h92: sine = 16'he4a4 ;
231                         8'h93: sine = 16'he33b ;
232                         8'h94: sine = 16'he1d6 ;
233                         8'h95: sine = 16'he075 ;
234                         8'h96: sine = 16'hdf1a ;
235                         8'h97: sine = 16'hddc4 ;
236                         8'h98: sine = 16'hdc73 ;
237                         8'h99: sine = 16'hdb27 ;
238                         8'h9a: sine = 16'hd9e1 ;
239                         8'h9b: sine = 16'hd8a1 ;
240                         8'h9c: sine = 16'hd767 ;
241                         8'h9d: sine = 16'hd634 ;
242                         8'h9e: sine = 16'hd506 ;
243                         8'h9f: sine = 16'hd3e0 ;
244                         8'ha0: sine = 16'hd2c0 ;
245                         8'ha1: sine = 16'hd1a7 ;
246                         8'ha2: sine = 16'hd095 ;
247                         8'ha3: sine = 16'hcf8b ;
248                         8'ha4: sine = 16'hce88 ;
249                         8'ha5: sine = 16'hcd8d ;
250                         8'ha6: sine = 16'hcc9a ;
251                         8'ha7: sine = 16'hcbae ;
252                         8'ha8: sine = 16'hcacb ;
253                         8'ha9: sine = 16'hc9ef ;
254                         8'haa: sine = 16'hc91c ;
255                         8'hab: sine = 16'hc852 ;
256                         8'hac: sine = 16'hc790 ;
257                         8'had: sine = 16'hc6d7 ;
258                         8'hae: sine = 16'hc626 ;
259                         8'haf: sine = 16'hc57f ;
260                         8'hb0: sine = 16'hc4e1 ;
261                         8'hb1: sine = 16'hc44b ;
262                         8'hb2: sine = 16'hc3bf ;
263                         8'hb3: sine = 16'hc33c ;
264                         8'hb4: sine = 16'hc2c3 ;
265                         8'hb5: sine = 16'hc253 ;
266                         8'hb6: sine = 16'hc1ec ;
267                         8'hb7: sine = 16'hc190 ;
268                         8'hb8: sine = 16'hc13c ;
269                         8'hb9: sine = 16'hc0f3 ;
270                         8'hba: sine = 16'hc0b3 ;
271                         8'hbb: sine = 16'hc07d ;
272                         8'hbc: sine = 16'hc050 ;
273                         8'hbd: sine = 16'hc02e ;
274                         8'hbe: sine = 16'hc015 ;
275                         8'hbf: sine = 16'hc006 ;
276                         8'hc0: sine = 16'hc001 ;
```

```
277                             8'hc1: sine = 16'hc006 ;
278                             8'hc2: sine = 16'hc015 ;
279                             8'hc3: sine = 16'hc02e ;
280                             8'hc4: sine = 16'hc050 ;
281                             8'hc5: sine = 16'hc07d ;
282                             8'hc6: sine = 16'hc0b3 ;
283                             8'hc7: sine = 16'hc0f3 ;
284                             8'hc8: sine = 16'hc13c ;
285                             8'hc9: sine = 16'hc190 ;
286                             8'hca: sine = 16'hc1ec ;
287                             8'hcb: sine = 16'hc253 ;
288                             8'hcc: sine = 16'hc2c3 ;
289                             8'hcd: sine = 16'hc33c ;
290                             8'hce: sine = 16'hc3bf ;
291                             8'hcf: sine = 16'hc44b ;
292                             8'hd0: sine = 16'hc4e1 ;
293                             8'hd1: sine = 16'hc57f ;
294                             8'hd2: sine = 16'hc626 ;
295                             8'hd3: sine = 16'hc6d7 ;
296                             8'hd4: sine = 16'hc790 ;
297                             8'hd5: sine = 16'hc852 ;
298                             8'hd6: sine = 16'hc91c ;
299                             8'hd7: sine = 16'hc9ef ;
300                             8'hd8: sine = 16'hcacb ;
301                             8'hd9: sine = 16'hcbae ;
302                             8'hda: sine = 16'hcc9a ;
303                             8'hdb: sine = 16'hcd8d ;
304                             8'hdc: sine = 16'hce88 ;
305                             8'hdd: sine = 16'hcf8b ;
306                             8'hde: sine = 16'hd095 ;
307                             8'hdf: sine = 16'hd1a7 ;
308                             8'he0: sine = 16'hd2c0 ;
309                             8'he1: sine = 16'hd3e0 ;
310                             8'he2: sine = 16'hd506 ;
311                             8'he3: sine = 16'hd634 ;
312                             8'he4: sine = 16'hd767 ;
313                             8'he5: sine = 16'hd8a1 ;
314                             8'he6: sine = 16'hd9e1 ;
315                             8'he7: sine = 16'hdb27 ;
316                             8'he8: sine = 16'hdc73 ;
317                             8'he9: sine = 16'hddc4 ;
318                             8'hea: sine = 16'hdf1a ;
319                             8'heb: sine = 16'he075 ;
320                             8'hec: sine = 16'he1d6 ;
321                             8'hed: sine = 16'he33b ;
322                             8'hee: sine = 16'he4a4 ;
323                             8'hef: sine = 16'he611 ;
324                             8'hf0: sine = 16'he783 ;
325                             8'hf1: sine = 16'he8f8 ;
326                             8'hf2: sine = 16'hea71 ;
327                             8'hf3: sine = 16'hebed ;
328                             8'hf4: sine = 16'hed6d ;
329                             8'hf5: sine = 16'heeef ;
330                             8'hf6: sine = 16'hf074 ;
331                             8'hf7: sine = 16'hf1fb ;
332                             8'hf8: sine = 16'hf384 ;
333                             8'hf9: sine = 16'hf510 ;
334                             8'hfa: sine = 16'hf69d ;
335                             8'hfb: sine = 16'hf82b ;
336                             8'hfc: sine = 16'hf9bb ;
337                             8'hfd: sine = 16'hfb4b ;
```

```
338                         8'hfe: sine = 16'hfcdd ;
339                         8'hff: sine = 16'hfe6e ;
340         endcase
341 end
342 endmodule
```

- **Output(229Hz using 1MHz clock)**

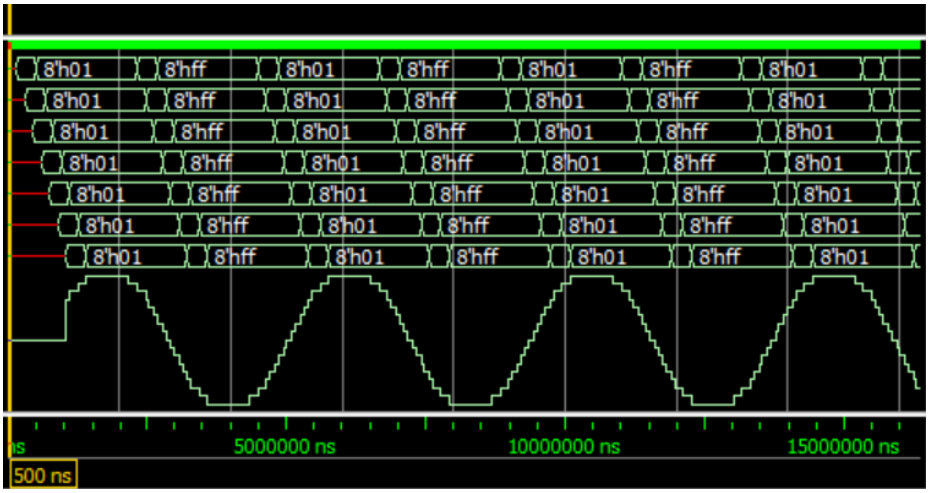


- Method 2

```
// this is calibration of waveform to real time
`timescale 100ns/100ns
module RisingEdge_DFlipFlop(D,clk,Q);// D = input, Q = output
  input signed [7:0] D;
  input clk;
   // Q is variable which stores output
  output reg signed [7:0] Q;
  //always at positive edge of the clock
   always @(posedge clk)
  Begin
          // Q becomes equal to D (property of D flip-flops)
          Q <= D;
  end
endmodule
//test bench module
module tb_DFF();
  reg [7:0] D;
  reg clk;
  wire [7:0] #1500 q1;
  wire [7:0] #1500 q2;
  wire [7:0] #1500 q3;
  wire [7:0] #1500 q4;
  wire [7:0] #1500 q5;
  wire [7:0] #1500 q6;
  wire [7:0] #1500 Q;
  wire [7:0] sin;
  // This sets the clock of 1MHz frequency
   initial begin
          clk=0;
```

```
            forever #5 clk = ~clk; //1MHz clock
    end
  // Always at positive edge of the clock this inputs are given
  to the circuit
    always @ (posedge clk) begin
            D <= 0;
            #3620;
            D <= 0.5;
            #3620;
            D <= 0.7071067;
            #3620;
            D <= 1;
            #3620;
            D <= 0.7071067;
            #3620;
            D <= 0.5;
            #3620;
            D <= 0;
            #3620;
            D <= -0.5;
            #3620;
            D <= -0.7071067;
            #3620;
            D <= -1;
            #3620;
            D <= -0.7071067;
            #3620;
            D <= -0.5;
            #3620;
            D <= 0;
    end
  //these are the functions which returns output of each DFF
    RisingEdge_DFlipFlop FF0(D,clk,q1);
    RisingEdge_DFlipFlop FF1(q1,clk,q2);
    RisingEdge_DFlipFlop FF2(q2,clk,q3);
    RisingEdge_DFlipFlop FF3(q3,clk,q4);
    RisingEdge_DFlipFlop FF4(q4,clk,q5);
    RisingEdge_DFlipFlop FF5(q5,clk,q6);
    RisingEdge_DFlipFlop FF6(q6,clk,Q);
  // Sum of all the outputs of DFF gives a sine wave
    assign sin = D+q1+q2+q3+q4+q5+q6+Q;
  endmodule
```

- **Output(230Hz using clock of 1MHz)**



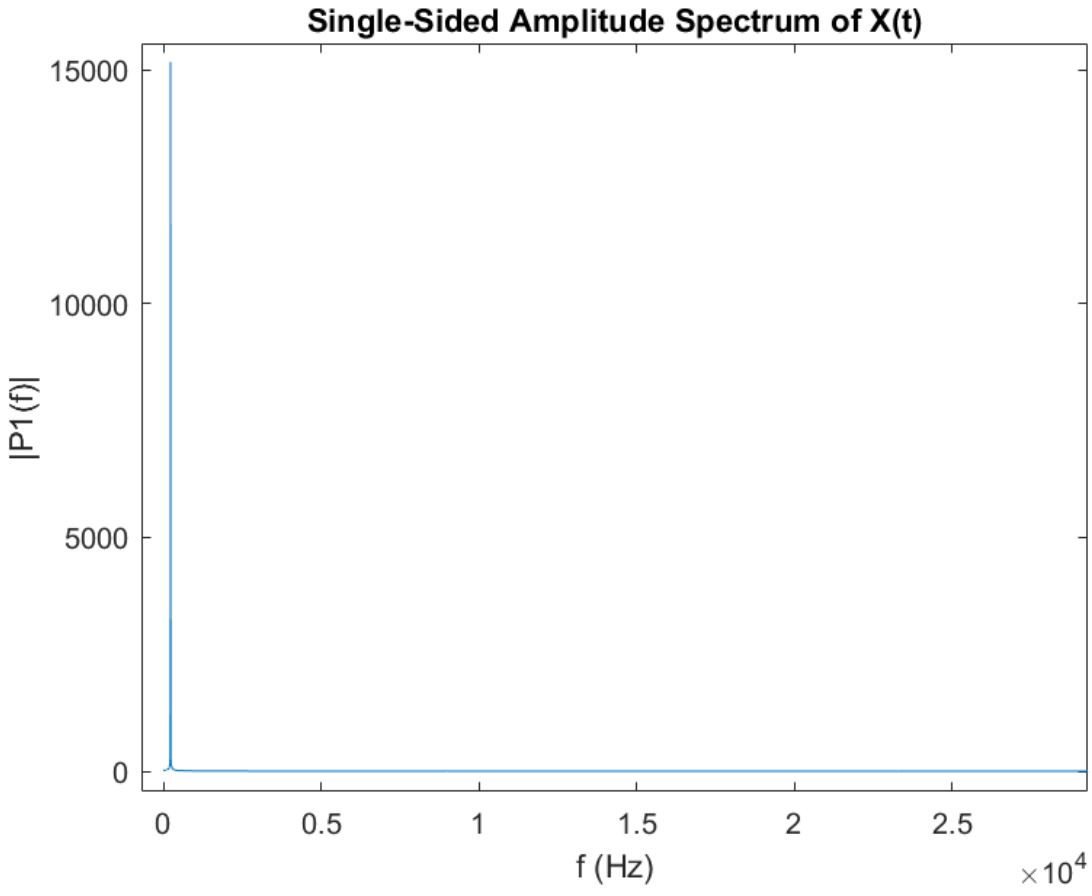
3. Purity in terms of harmonic distortion

Formula to calculate Purity of a signal in terms of Total Harmonic Distortion (THD) using FFT of the signal is given by:

$$\text{THD (w)} = \frac{\sqrt{\sum_{n=2}^{10} Y_n^2(w)}}{Y_1}.100\%$$

Where, Y_i is the magnitude of the peak.

For Method 1:

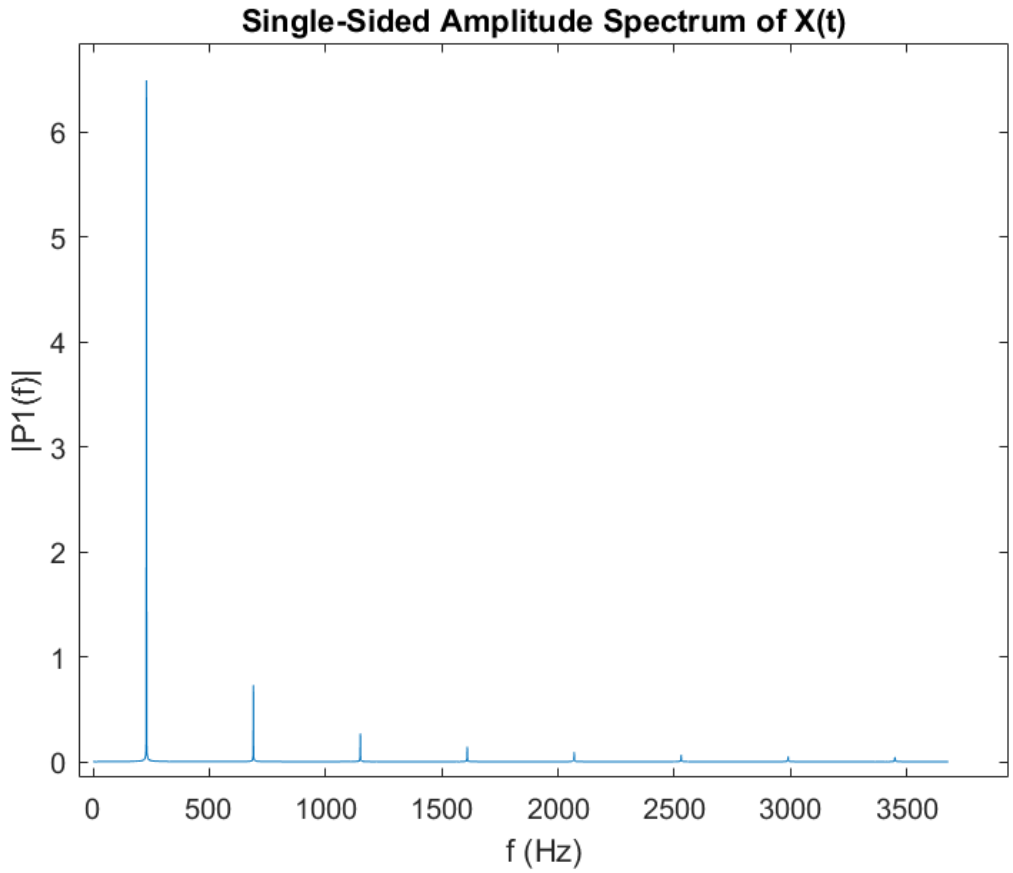


Peak Number	Magnitude	Frequency(in Hz)
1	15992	229

In this case since there is only one peak, THD = 0%

Thus this is a pure signal.

For Method 2:



Peak Number	Magnitude	Frequency(in Hz)
1	6.495	230
2	0.731	690
3	0.269	1150
4	0.143	1610
5	0.091	2070
6	0.051	2990
7	0.042	3450

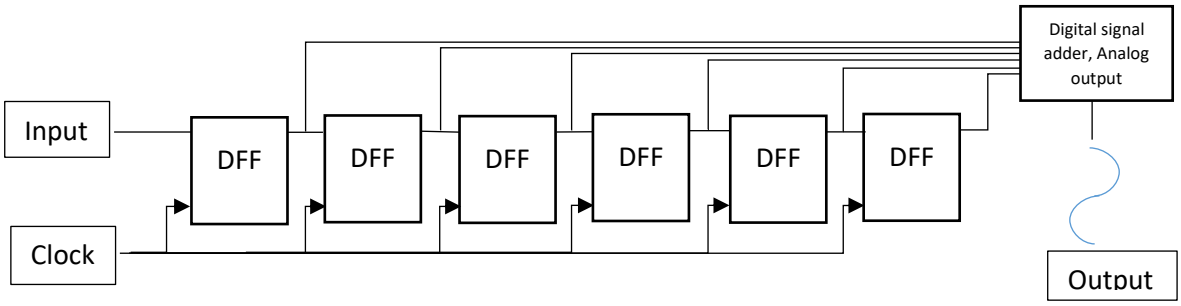
So, THD = (0.79988/6.495) x 100 % = 12.3%

Hence, it is not a pure signal. This signal is distorted from the pure sine signal by 12.3%.

MATLAB codes for the above FFT analysis and ModelSim projects can be found in the GitHub link provided below.

https://github.com/zorawar12/Digital-Circuits-and-Systems

4. Circuit design description for Method 2



This circuit consists of six D-Flip Flops in series. When an input is given to this circuit, the output from each of the flip-flops will be delayed by some time due to the long wires in this case. Summing the output of each DFF will result into a sine wave as depicted below.

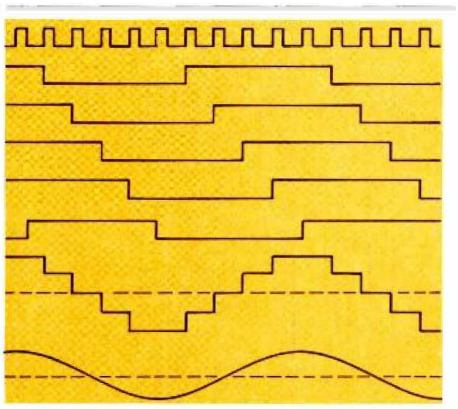
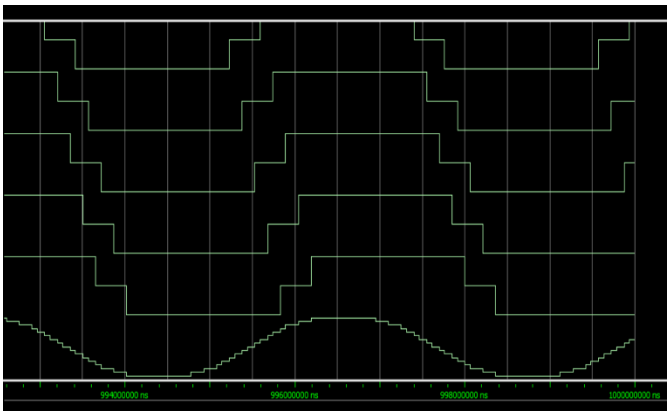


Image courtesy: www.tinaja.com



Result expanded from output of Method 2

The number of inputs at every positive edge of the clock and the time delay between two consecutive inputs and the delay in input to each DFF decides the final frequency and purity of the output waveform.

5. Comparison of two methods

Method 1 resulted into a more pure sine wave than Method 2. The reasons could be:

- In method 1, we can decide the number of sinewave values to be stored in memory (which is the lookup table) for a given frequency. Based on that, actual sine values are being thrown out at particular rate.
- In method 2, very few sine wave values has been used. And the output is not just a sine value rather it's a sum of outputs of DFF running continuously at certain rate. So, its purity can be increased only by increasing the number of DFF and increasing the number of inputs of actual sine values.
- In other words, the first method is where one uses DDS method where frequency of wave depends on reference-clock frequency and the binary number programmed into the frequency register which are accessed through a phase accumulator (a modulo-M counter) whereas second uses D-flip-flops. Because in DDS method, one uses the systems base clock and the period of the waveform is controlled by the frequency control register and finally the digital signal is converted to an analogue signal, it gives a pure output.
- So concluding from the overall working of the two methods, we can say that both the approaches are same but are different from the perspective of their sophistication and hardware requirements and hence are having difference in purity.

6. References

1. https://people.ece.cornell.edu/land/courses/ece5760/ModelSim/index.html
2. https://www.tinaja.com/glib/rad_elec/digital_sinewaves_11_76.pdf
3. https://people.ece.cornell.edu/land/courses/ece5760/ModelSim/testbench.v
4. https://www.gamry.com/application-notes/EIS/total-harmonic-distortion/
5. https://www.dataq.com/data-acquisition/general-education-tutorials/fft-fast-fourier-transform-waveform-analysis.html
6. https://in.mathworks.com/help/matlab/ref/fft.html
7. https://www.analog.com/en/analog-dialogue/articles/all-about-direct-digital-synthesis.html#