

EXPERIMENT 1

Full Name: Mohammad Taghizadeh

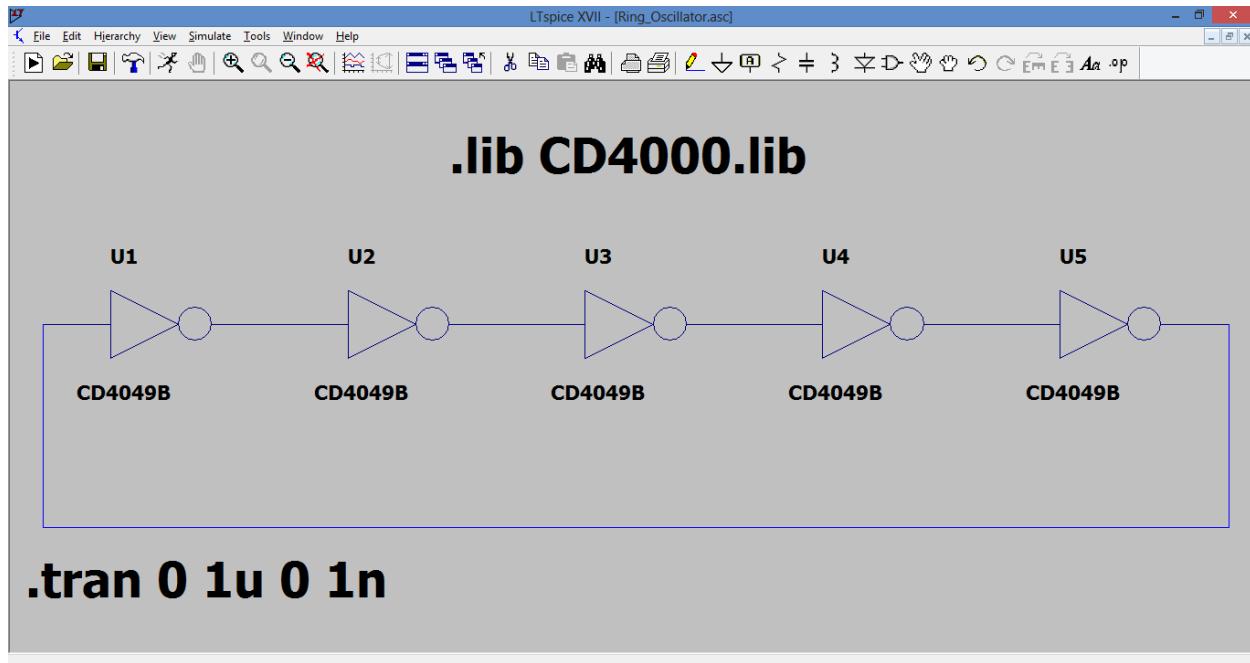
Student ID: 810198373

Contents

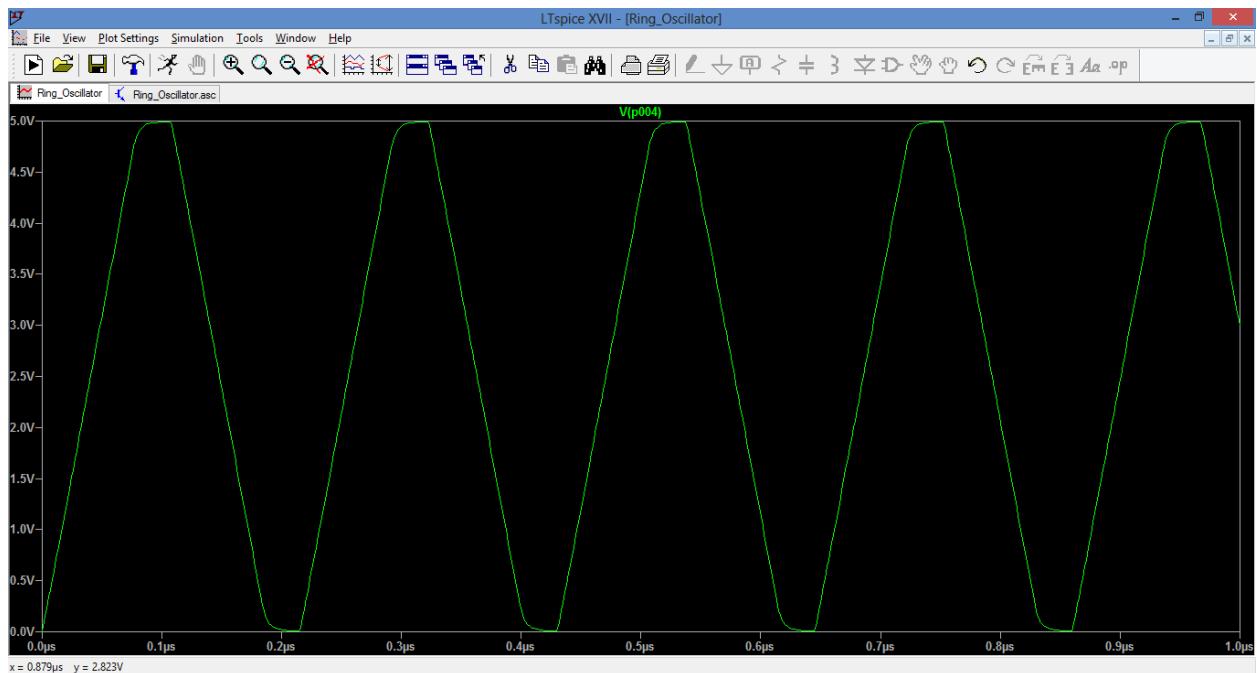
Ring Oscillator	2
Two Phase Clock Generator	4
LM555	5
Schmitt Trigger Oscillator	10
Ring Oscillator(Part 2 - FPGA design)	12
Frequency divider	13
Baud Rate Generator for UART Serial Communication	18

Ring Oscillator

Circuit:



Simulation of the above Ring oscillator's output:



N = number of inverters used in the Ring Oscillator = 5

Output's period time = 215 ns

1.

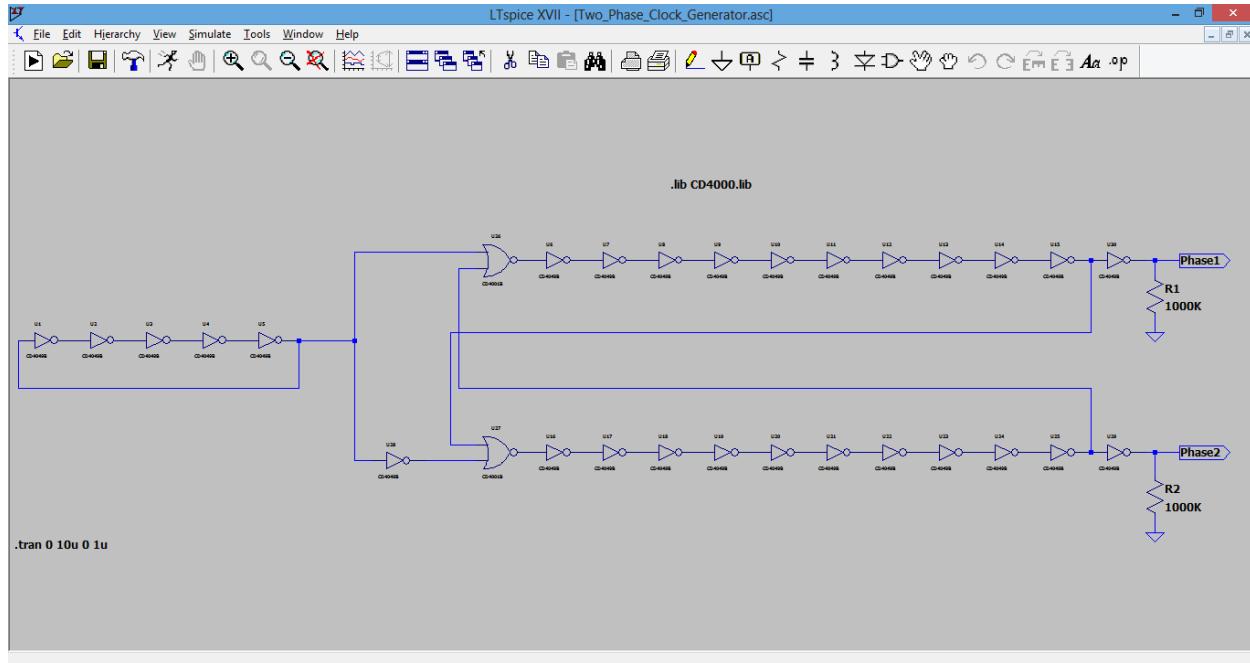
Propagation delay = $N * \text{delay of an inverter} = \text{output's period time} / 2 = 107.69 \text{ ns}$

2.

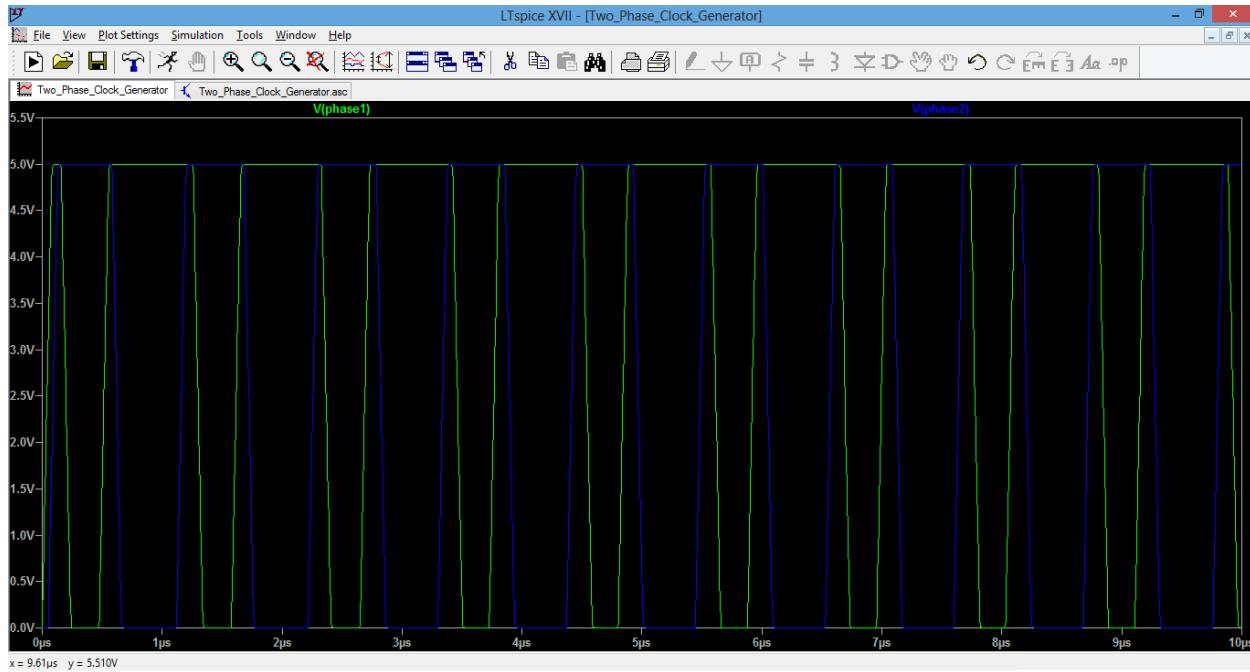
Single inverter's delay = $\text{output's period time} / N = \text{output's period time} / 5 = 21 \text{ ns}$

Two Phase Clock Generator

Circuit:

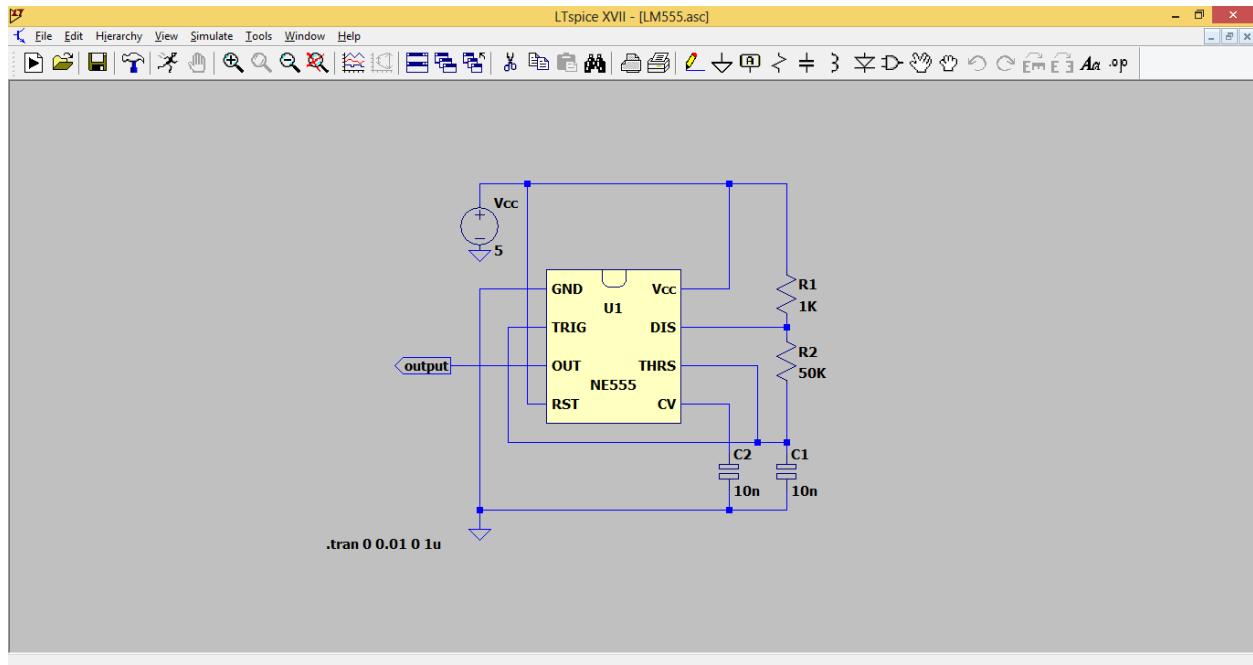


Simulation of the above Two Phase Clock Generator's two phases output:

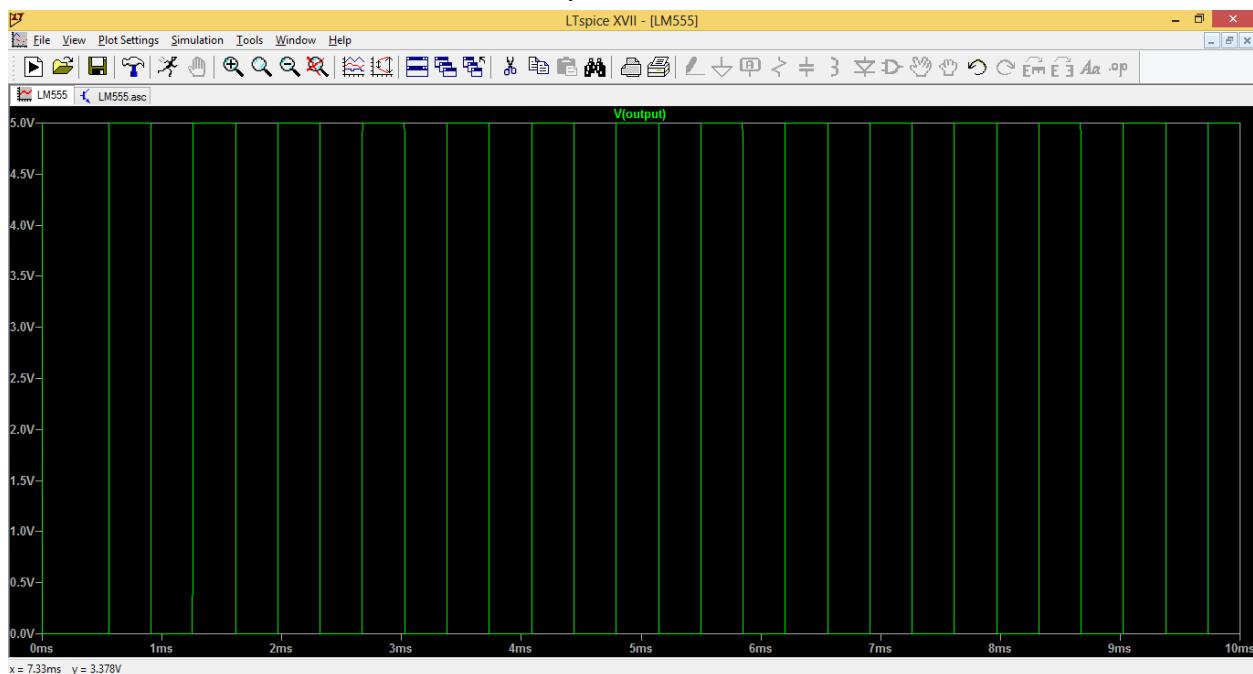


LM555

Circuit:



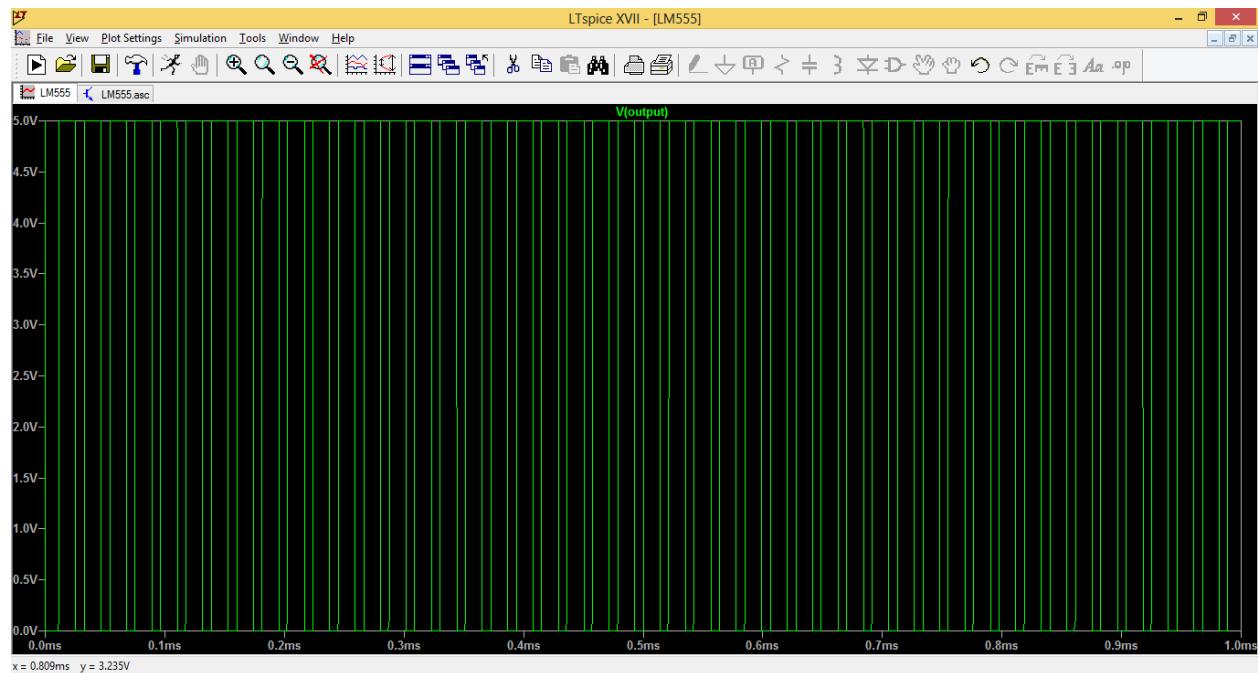
Simulation of the above LM555's output:



Clock frequency = 1428.7143, Duty cycle = 0.50495 \approx 0.5

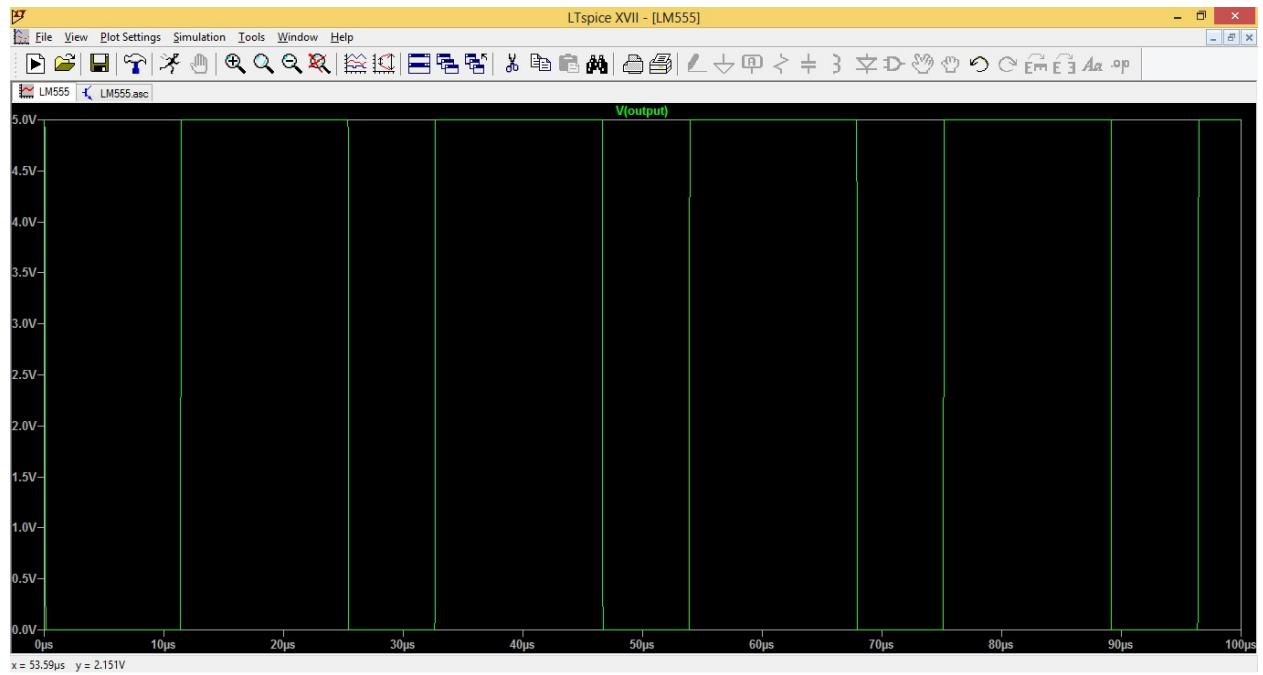
2.

$$R2 = 1\text{K}\Omega \Rightarrow \text{frequency} = 48100.048, \text{ duty cycle} = 67\%$$

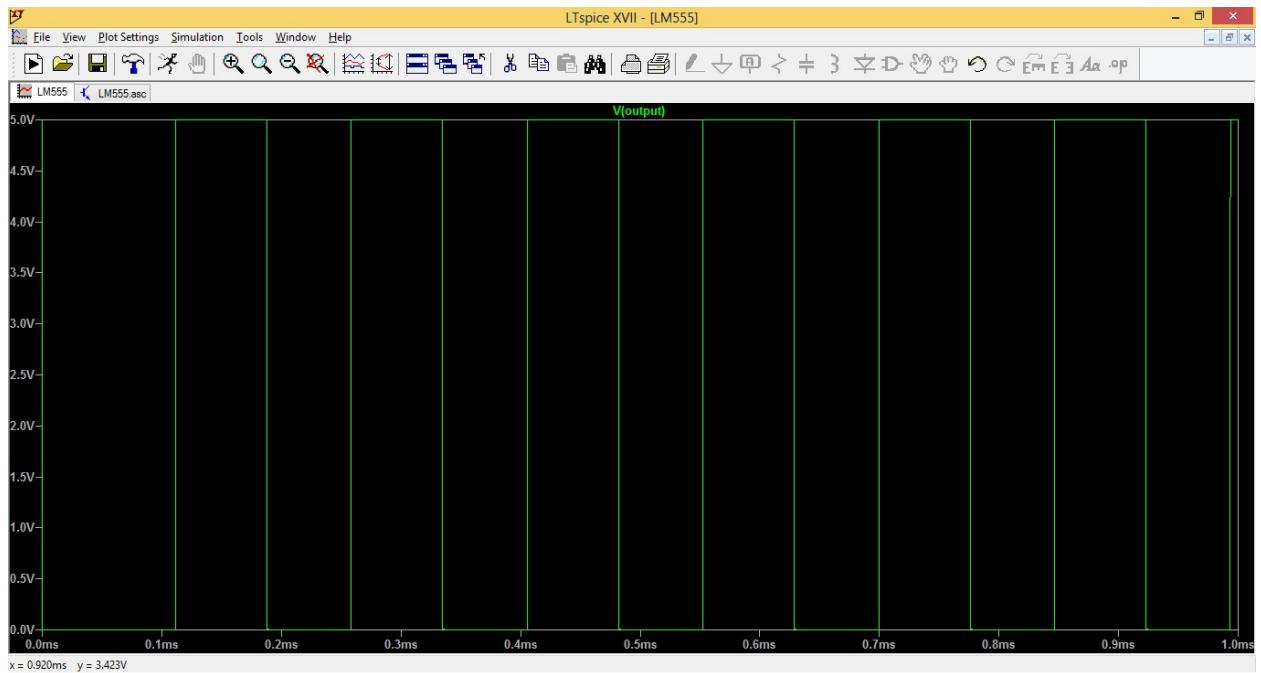


As you see in above diagram, in 0.1ms we almost have 4 periods, so in 1s we will almost have 40000 periods which is close to 48100.048.

As you see in below diagram, duty cycle is almost 2/3 which is close to 67%.

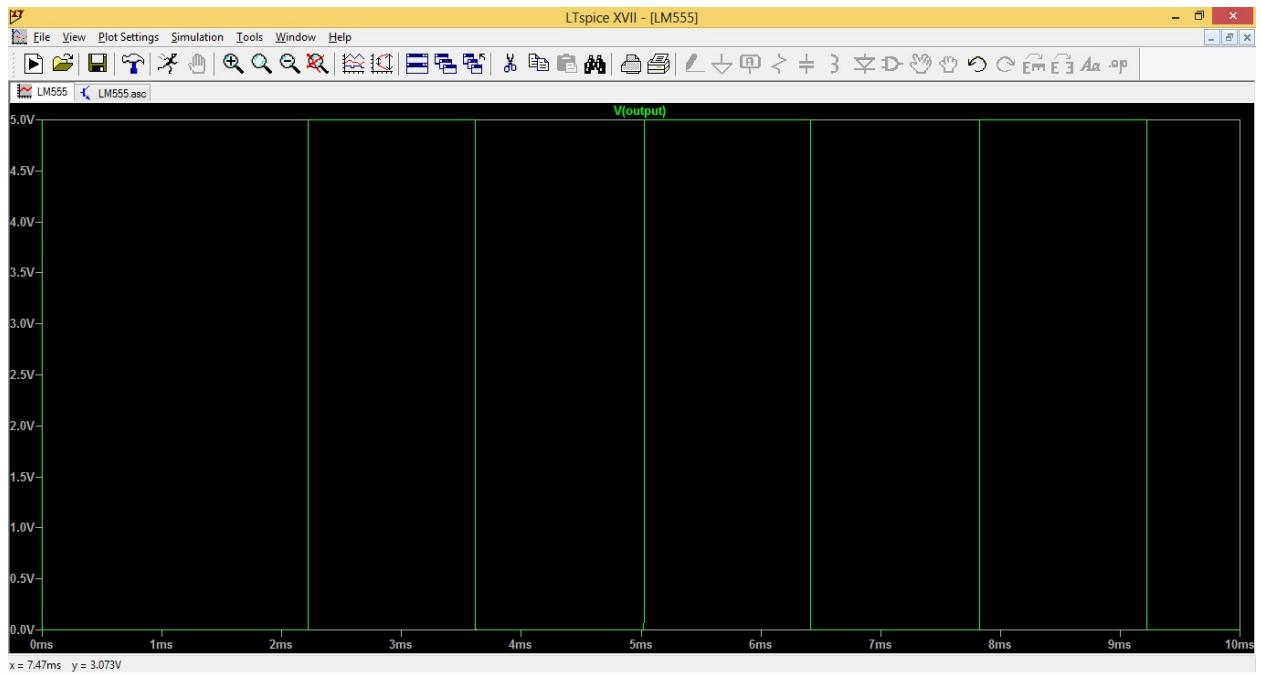


$R_2 = 10K\Omega \Rightarrow \text{frequency} = 6871.435, \text{ duty cycle} = 52.38\%$



As you see in above diagram, in 1ms we almost have 6 periods, so in 1s we will almost have 6000 periods which is close to 6871.435 and duty cycle is near 50% which is close to 52.38%.

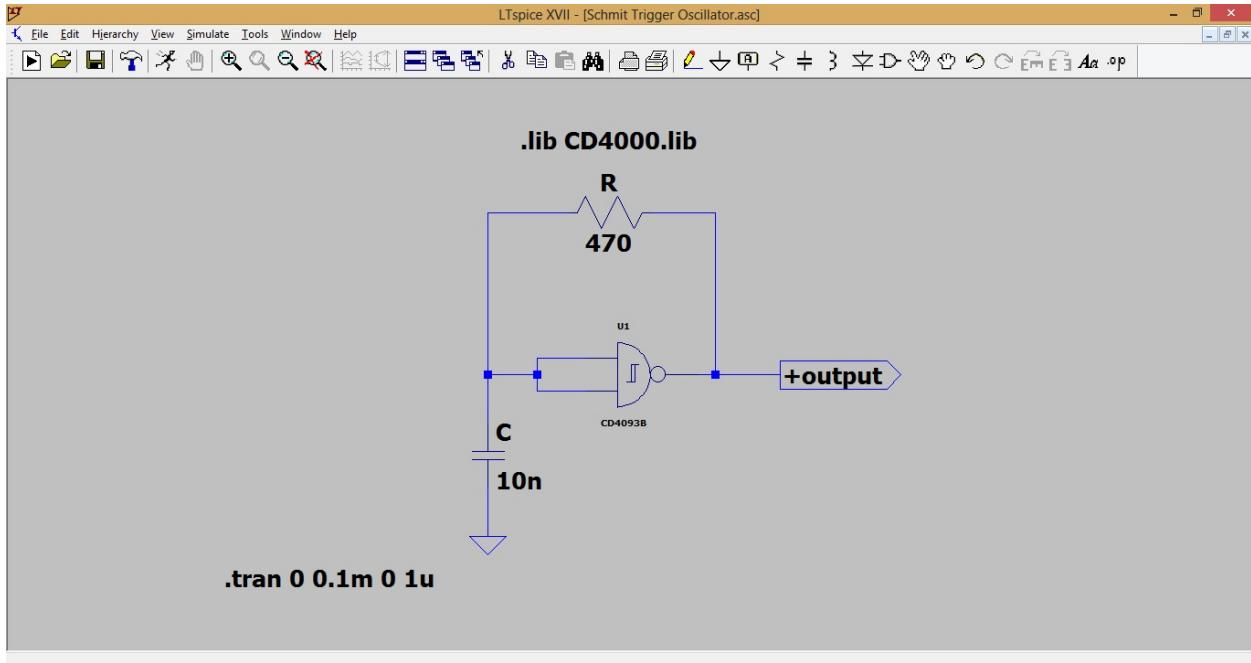
$R_2 = 200\text{ k}\Omega \Rightarrow \text{frequency} = 359.85$, duty cycle = 50.12%



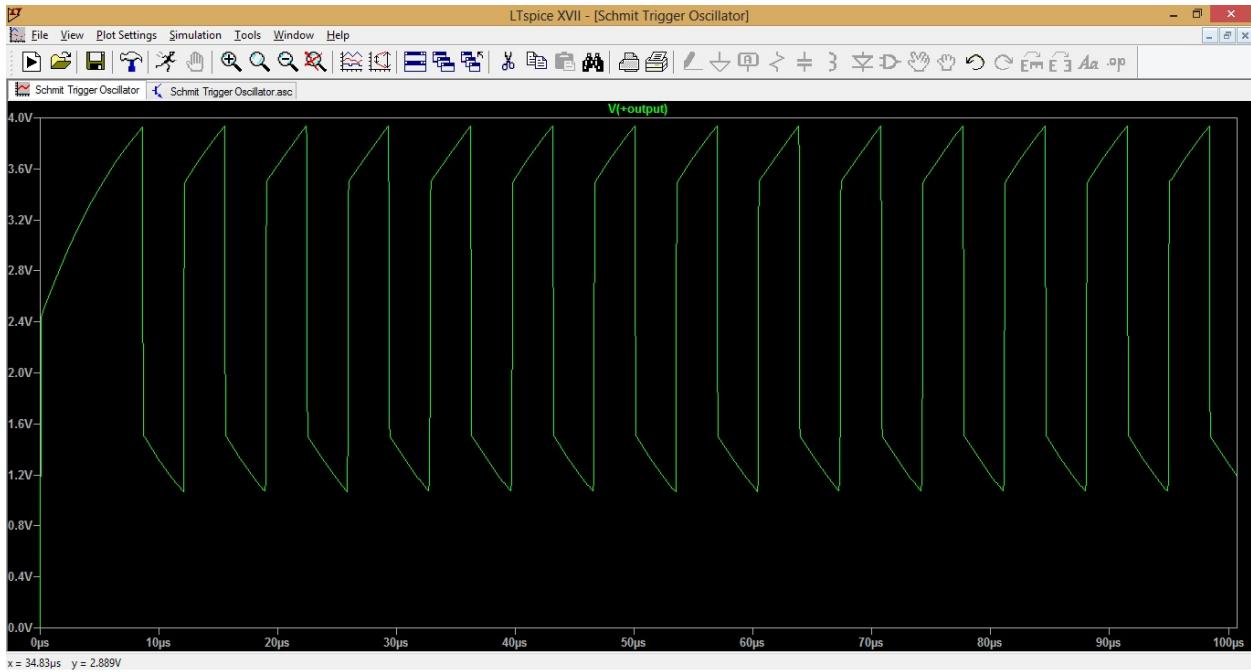
As you see in above diagram, in 10ms we almost have 3 periods, so in 1s we will almost have 300 periods which is close to 359.85 and duty cycle is near 50% which is close to 50.12%.

Schmitt Trigger Oscillator

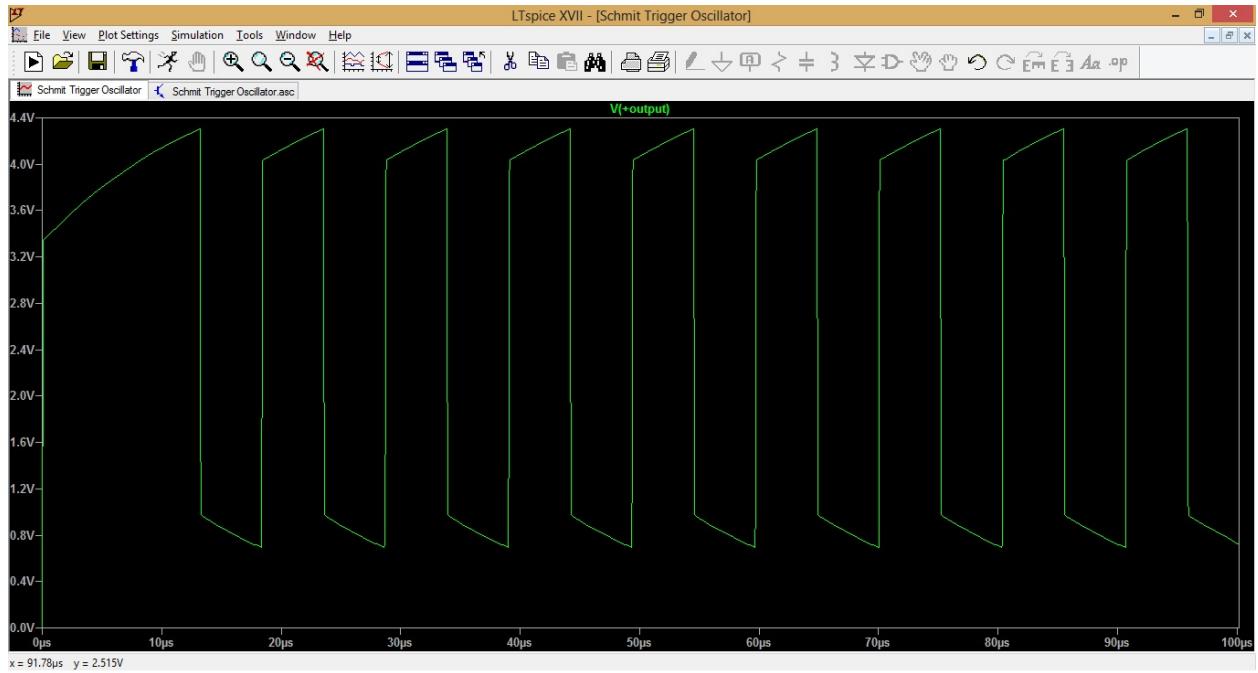
Circuit:



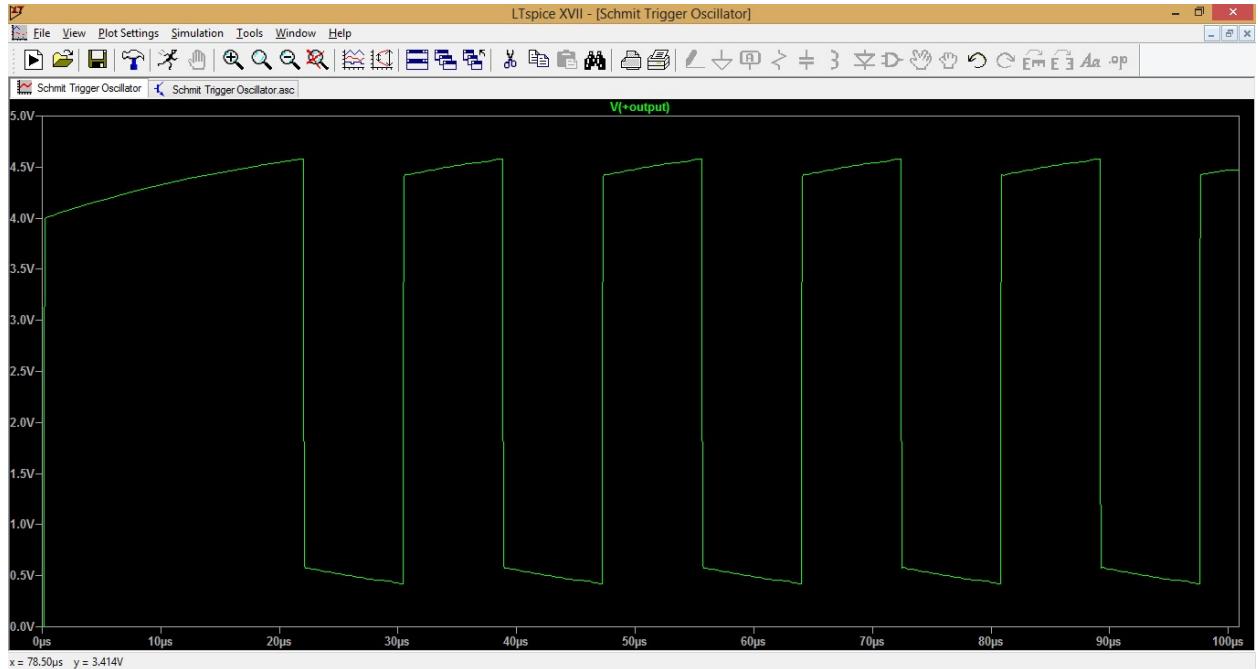
Simulation of the above Schmitt Trigger Oscillator's output for $R = 470\Omega$:



Simulation of the Schmitt Trigger Oscillator's output for $R = 1000\Omega$:



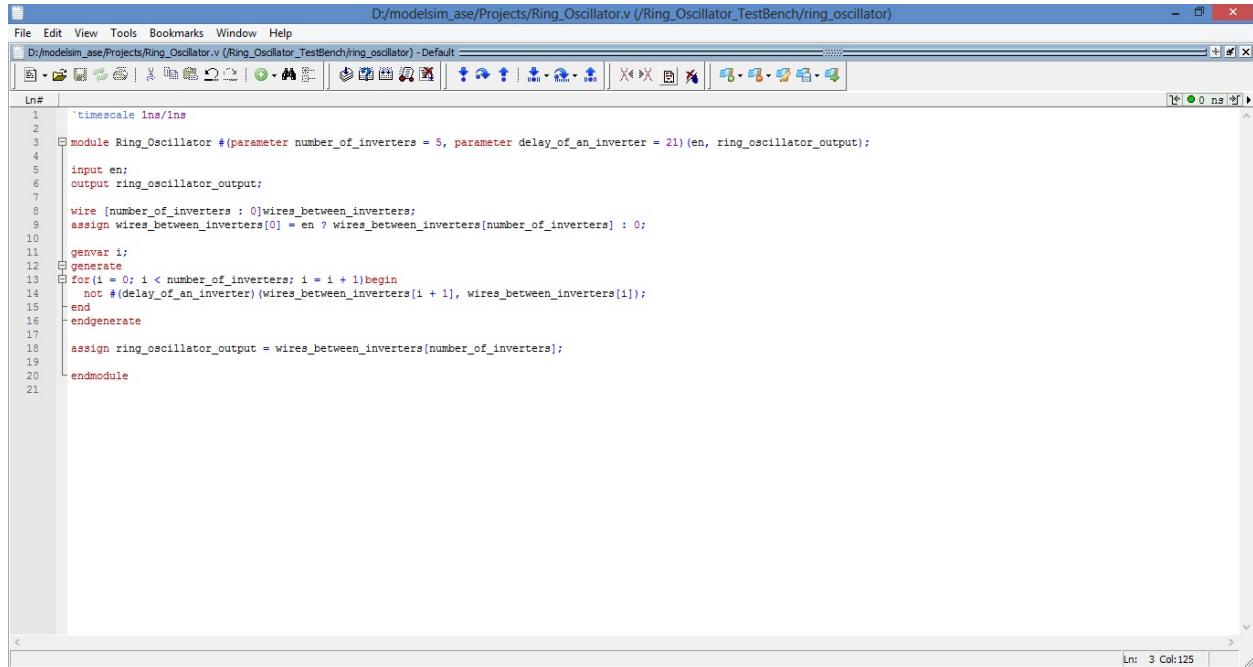
Simulation of the Schmitt Trigger Oscillator's output for $R = 2000\Omega$:



$$\alpha_{2000} = 1.195, \alpha_{1000} = 0.974, \alpha_{470} = 0.6867 \Rightarrow \alpha_{\text{average}} = 0.952$$

Ring Oscillator(Part 2 - FPGA design)

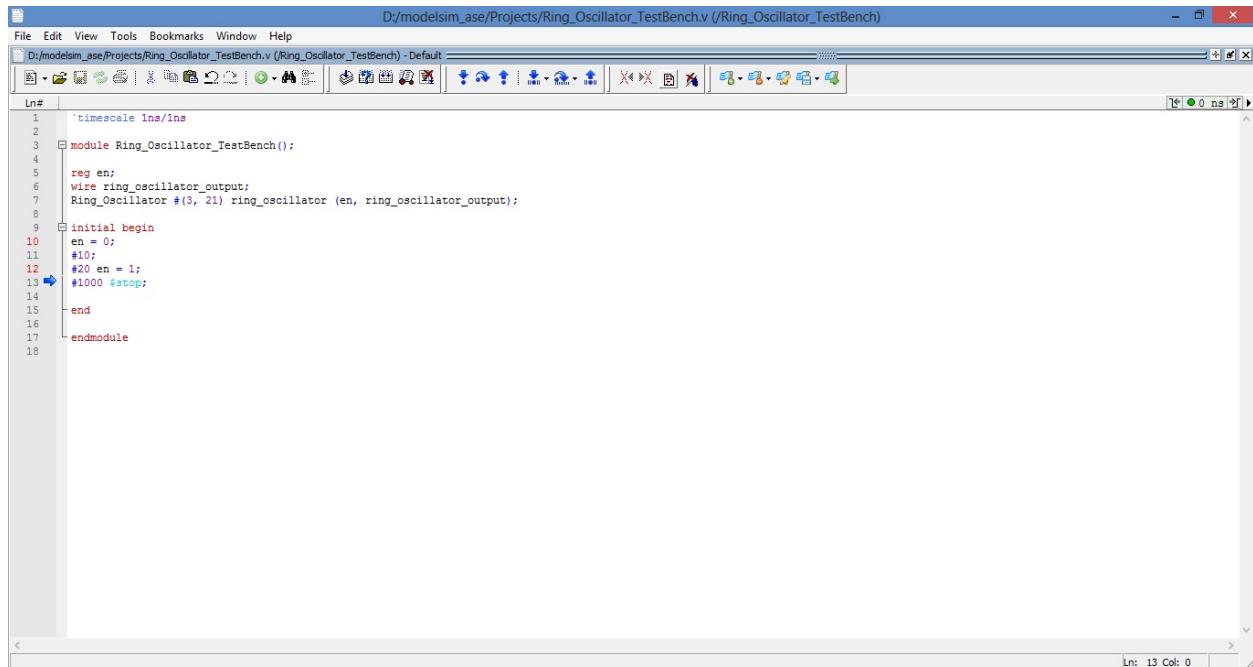
Verilog description of Ring Oscillator circuit:



The screenshot shows the ModelSim ASE interface with the Verilog code for the Ring Oscillator module. The code defines a module named 'Ring_Oscillator' with parameters for the number of inverters (5) and delay of an inverter (21). It has an input 'en' and an output 'ring_oscillator_output'. Inside the module, a wire 'wires_between_inverters' is initialized to 0. A generate block creates 5 inverters, each with a delay of 21. The output of the last inverter connects back to the first. The final assignment sets 'ring_oscillator_output' to the value of the last wire.

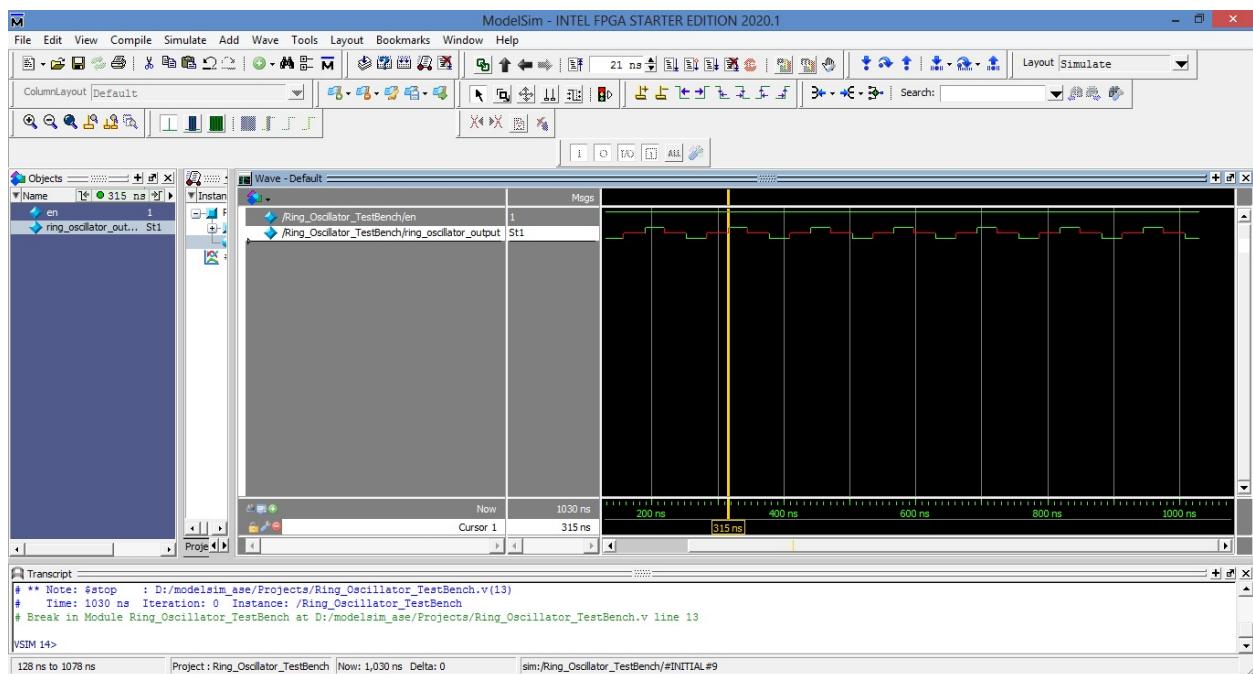
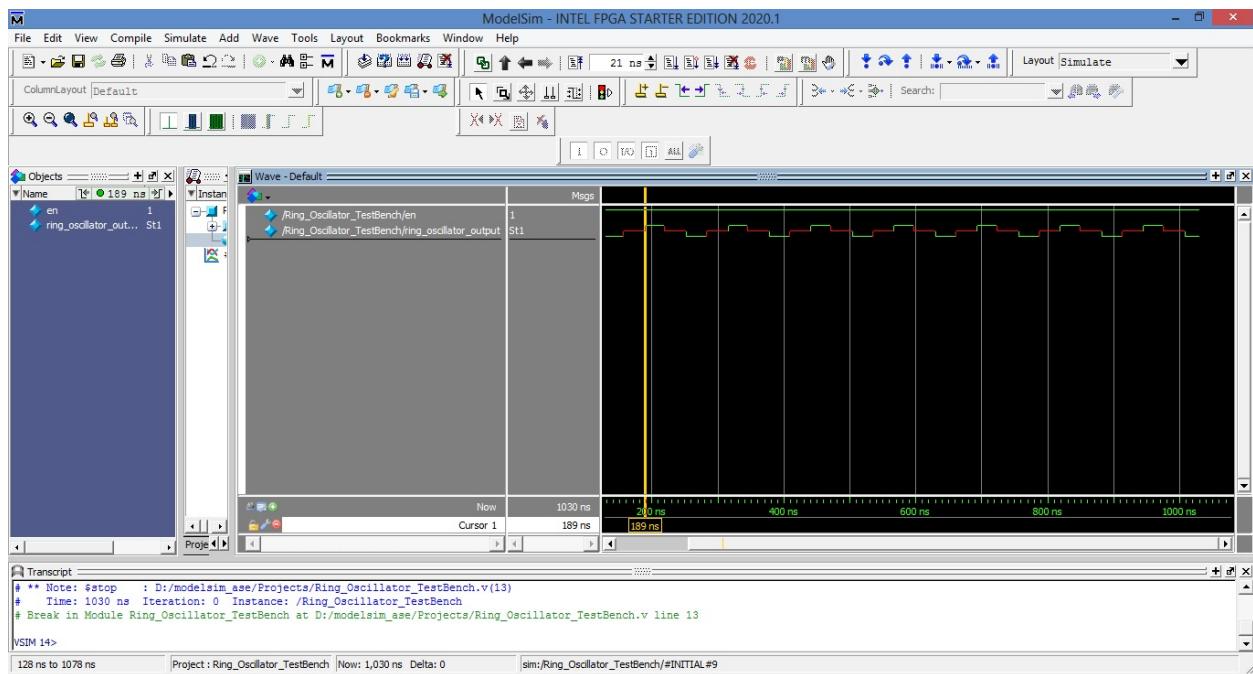
```
1 `timescale 1ns/1ns
2
3 module Ring_Oscillator #(parameter number_of_inverters = 5, parameter delay_of_an_inverter = 21) (en, ring_oscillator_output);
4   input en;
5   output ring_oscillator_output;
6
7   wire [number_of_inverters : 0] wires_between_inverters;
8   assign wires_between_inverters[0] = en ? wires_between_inverters[number_of_inverters] : 0;
9
10  genvar i;
11  generate
12    for(i = 0; i < number_of_inverters; i = i + 1)begin
13      nos #(delay_of_an_inverter) (wires_between_inverters[i + 1], wires_between_inverters[i]);
14    end
15  endgenerate
16
17  assign ring_oscillator_output = wires_between_inverters[number_of_inverters];
18
19 endmodule
20
21
```

Test Bench of above Ring Oscillator Verilog code:



The screenshot shows the ModelSim ASE interface with the Verilog code for the Ring Oscillator Test Bench. It includes an initial block that initializes the enable signal 'en' to 0, waits 10 units of time, then sets it to 1, and stops the simulation after 1000 units of time. The test bench then instantiates the 'Ring_Oscillator' module with 3 inverters and a delay of 21.

```
1 `timescale 1ns/1ns
2
3 module Ring_Oscillator_TestBench();
4
5   reg en;
6   wire ring_oscillator_output;
7   Ring_Oscillator #(3, 21) ring_oscillator (en, ring_oscillator_output);
8
9   initial begin
10     en = 0;
11     #10;
12     #20 en = 1;
13     #1000 #stop;
14   end
15
16 endmodule
17
18
```



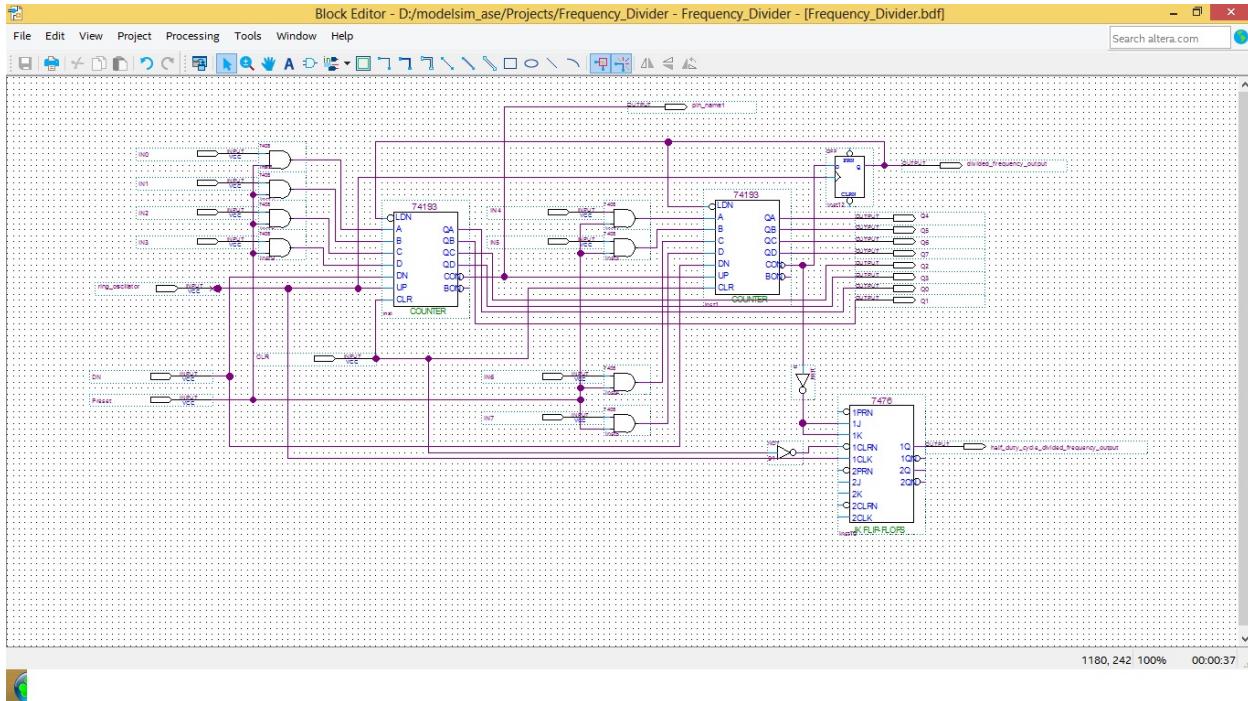
Given N in test bench = 3, given inverter's delay in test bench = 21ns

$$T = 2 \times N \times \text{inverter's delay} = 2 \times (3) \times (21\text{ns}) = 6 \times 21 = 126\text{ns} = 315 - 189$$

$$\Rightarrow f = 1/T = 1/126\text{ns} = 7,936,508 \text{ Hz}$$

Frequency divider

Frequency divider's circuit:



Verilog output of above Frequency divider's circuit:

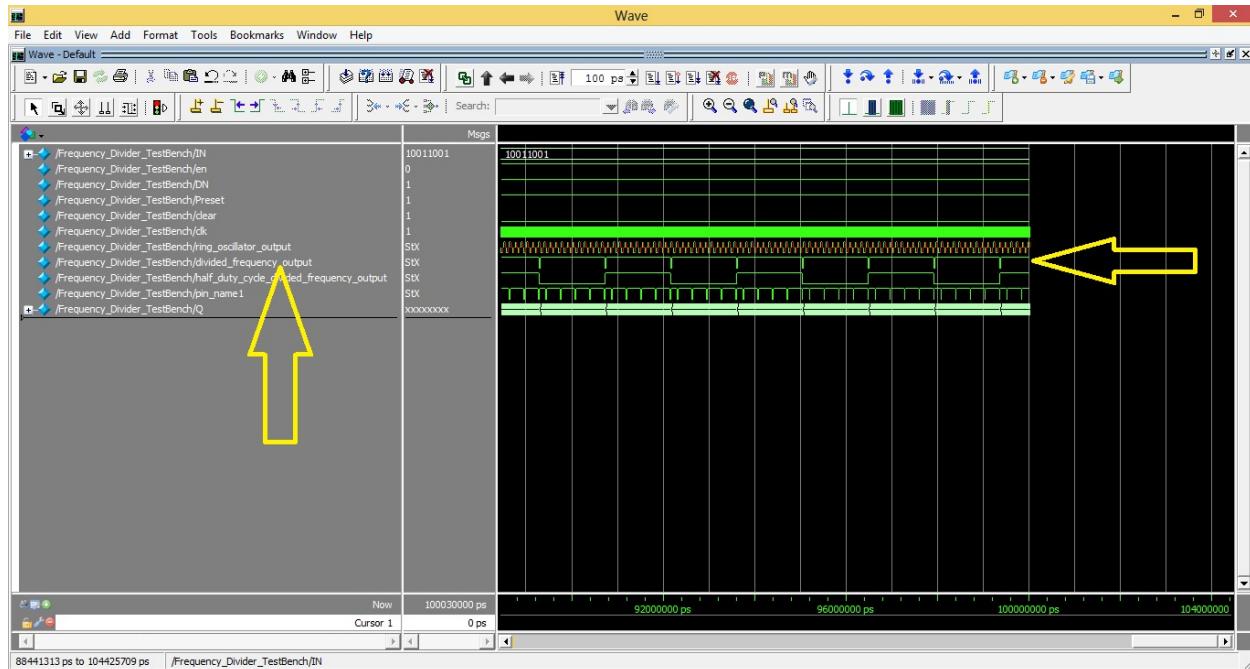
```
19 // DATE "04/20/2022 11:12:55"
20
21
22 // Device: Altera EP4CE6E22C6 Package TQFP144
23 //
24
25
26 // This Verilog file should be used for ModelSim-Altera (Verilog) only
27 //
28
29
30 `timescale 1 ns/ 1 ns
31
32 module Frequency_Divider (
33     Q7,
34     Preset,
35     INT,
36     IN6,
37     IN4,
38     CLR,
39     IN3,
40     IN2,
41     IN1,
42     IN0,
43     ring_oscillator,
44     DN,
45     enable,
46     Q6,
47     Q5,
48     Q4,
49     Q0,
50     Q1,
51     Q3,
52     Q2,
53     divided_frequency_output,
54     pin_name1,
55     IN5);
56     output Q7;
57
58 endmodule
```

Test Bench of Frequency divider's Verilog output:

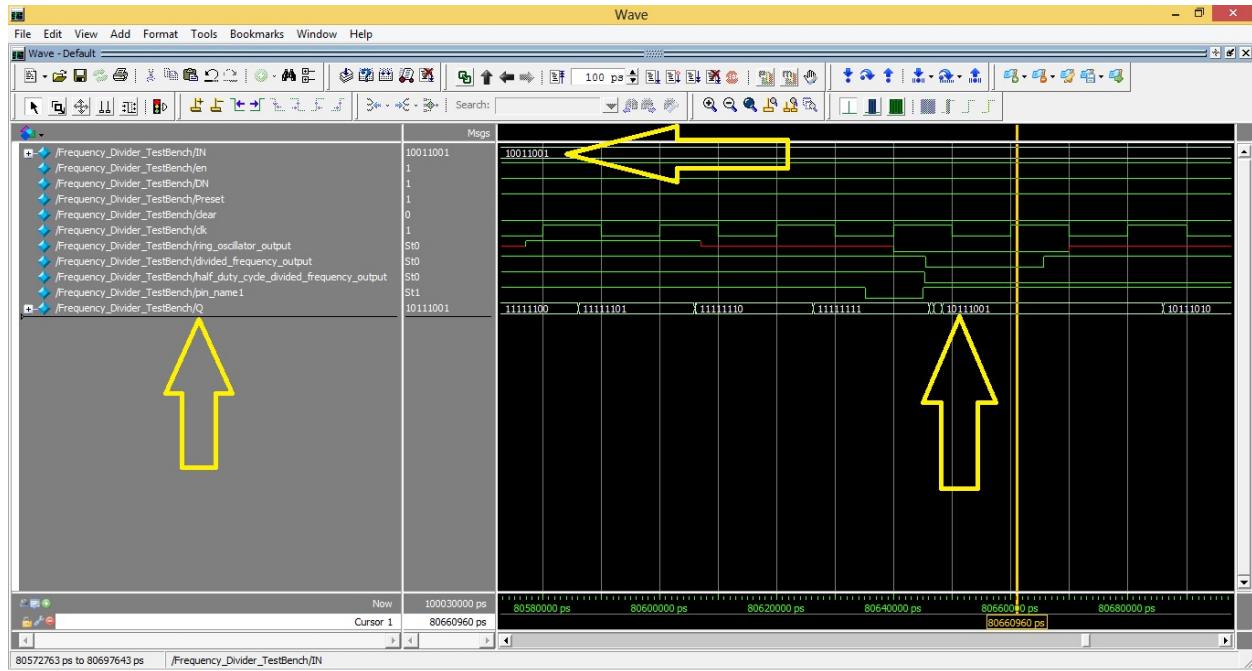
The screenshot shows the ModelSim interface with the Verilog code for the Frequency Divider Test Bench. The code defines a module with various inputs (IN, DN, Preset, clear, clk) and outputs (ring_oscillator_output, divided_frequency_output, half_duty_cycle_divided_frequency_output, pin_name1, Q). It includes a RING_OSCILLATOR component and a testbench logic. A yellow arrow points to the \$stop command at line 21.

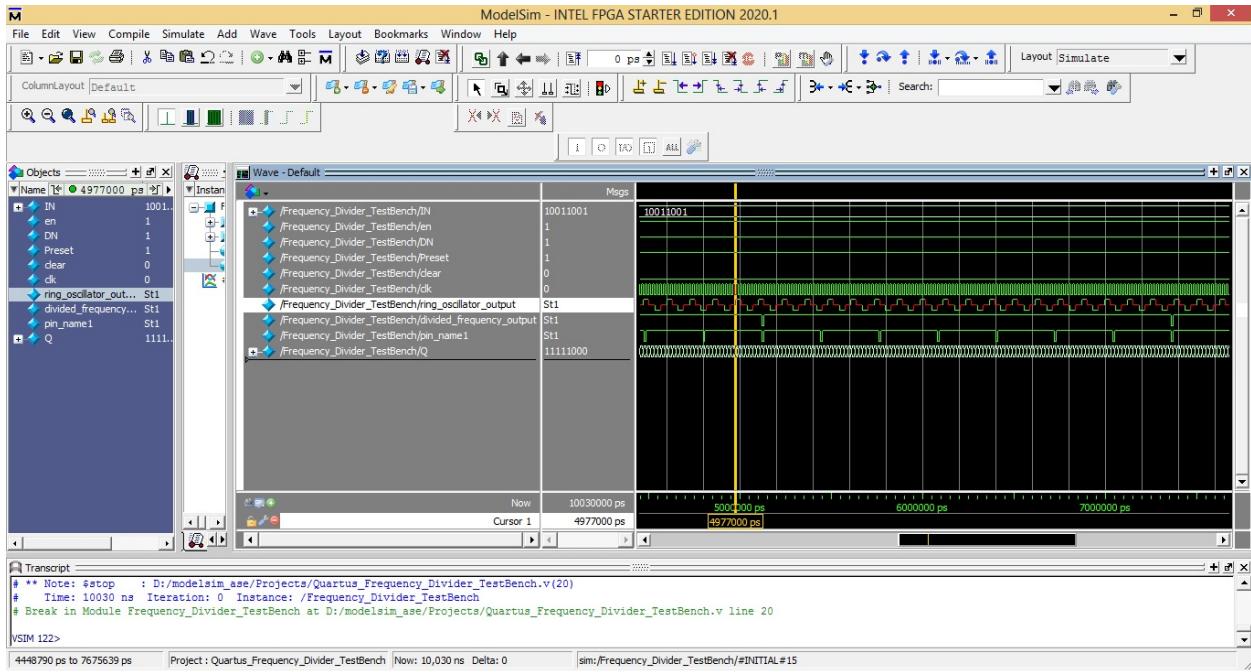
```
1 `timescale 1 ns/ 1 ns
2
3 module Frequency_Divider_TestBench();
4
5 reg [7 : 0]IN;
6 reg en, DN, Preset, clear, clk;
7 wire ring_oscillator_output, divided_frequency_output, half_duty_cycle_divided_frequency_output, pin_name1;
8 wire [7 : 0]Q;
9
10 RING_Oscillator #(8, 21) ring_oscillator(en, ring_oscillator_output);
11 Frequency_Divider frequency_divider(Q[7], Preset, IN[7], IN[6], IN[4], clear, IN[3], IN[2], IN[1], IN[0], clk, DN, Q[6], Q[5], Q[4], Q[3], Q[2],
12 divided_frequency_output, pin_name1,
13 half_duty_cycle_divided_frequency_output, IN[5]);
14
15 always #10 clk = ~clk;
16 initial begin
17 clk = 1;
18 IN = 8'b10011001; en = 0; clear = 1; Preset = 1; DN = 1;
19 #10 clear = 0;
20 #20 en = 1;
21 #1000000 $stop;
22 end
23 endmodule
```

Simulation of the above Frequency divider's output:



Loading the initial value (256 - 103) of Frequency divider:

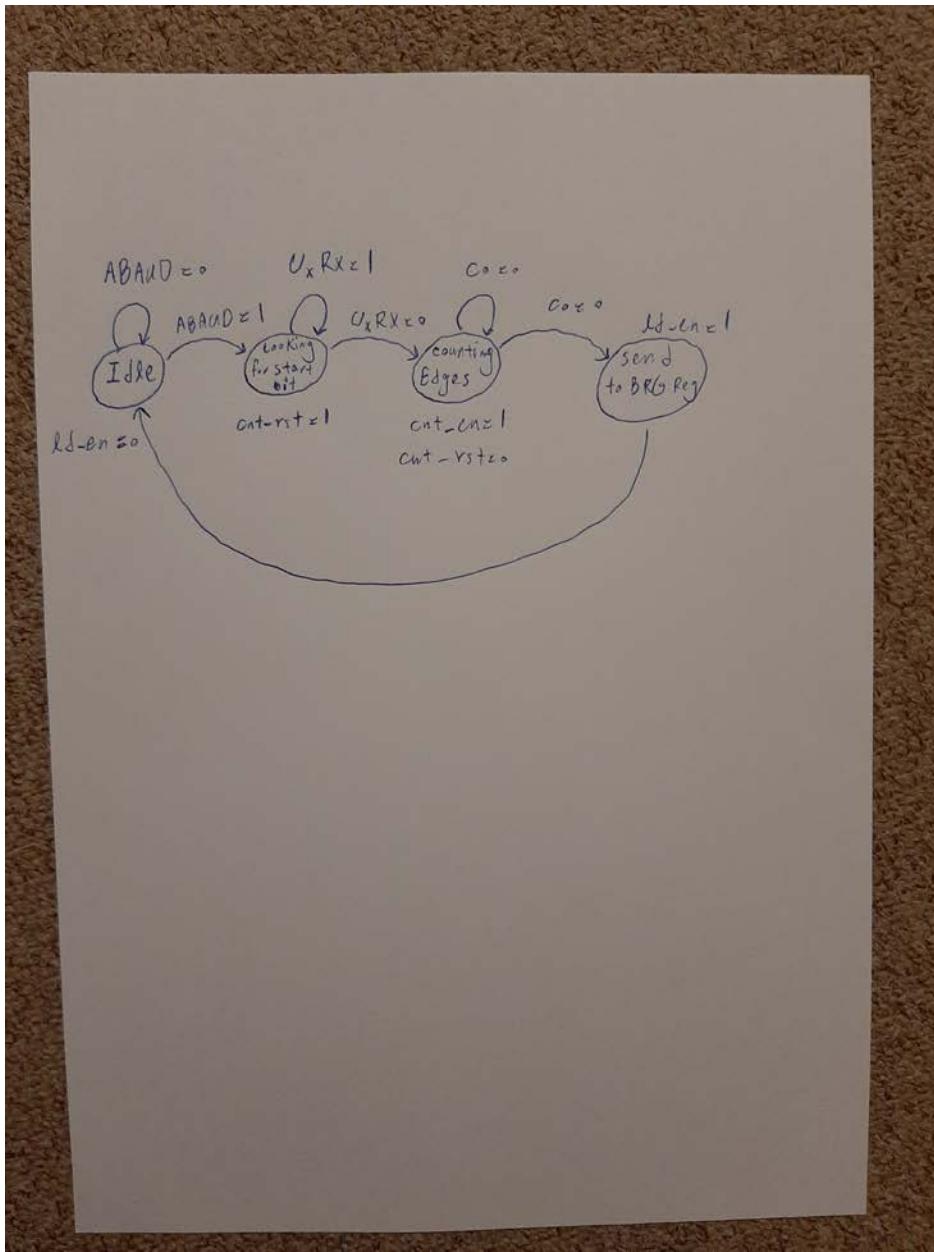




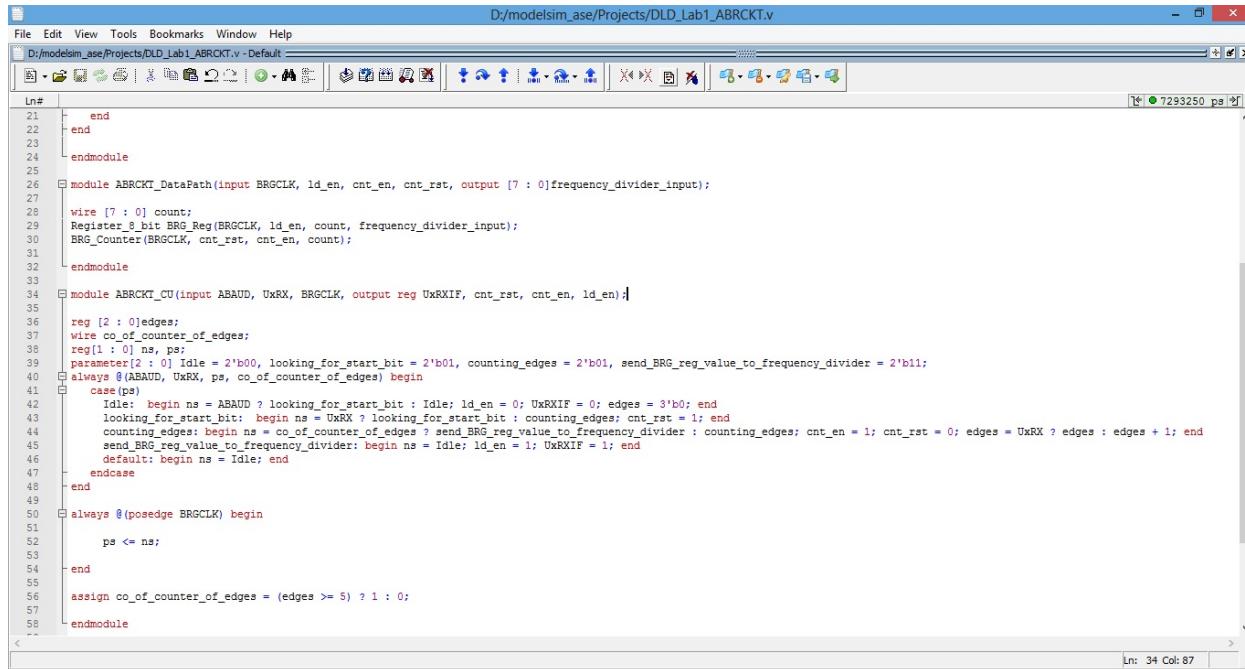
As you see in above diagram there are 7 periods between 2 periods of divided_frequency_output signal. Each period of pin_name1 signal is equal to 16 periods of clock, so period of divided_frequency_output is almost 16×7 clock period = $102 \times$ clock period, which is very close to $103 \times$ clock period.

Frequency = $1 / \text{period} = 1 / (103 \times \text{clock period}) = 1/103$ clock frequency

Baud Rate Generator for UART Serial Communication



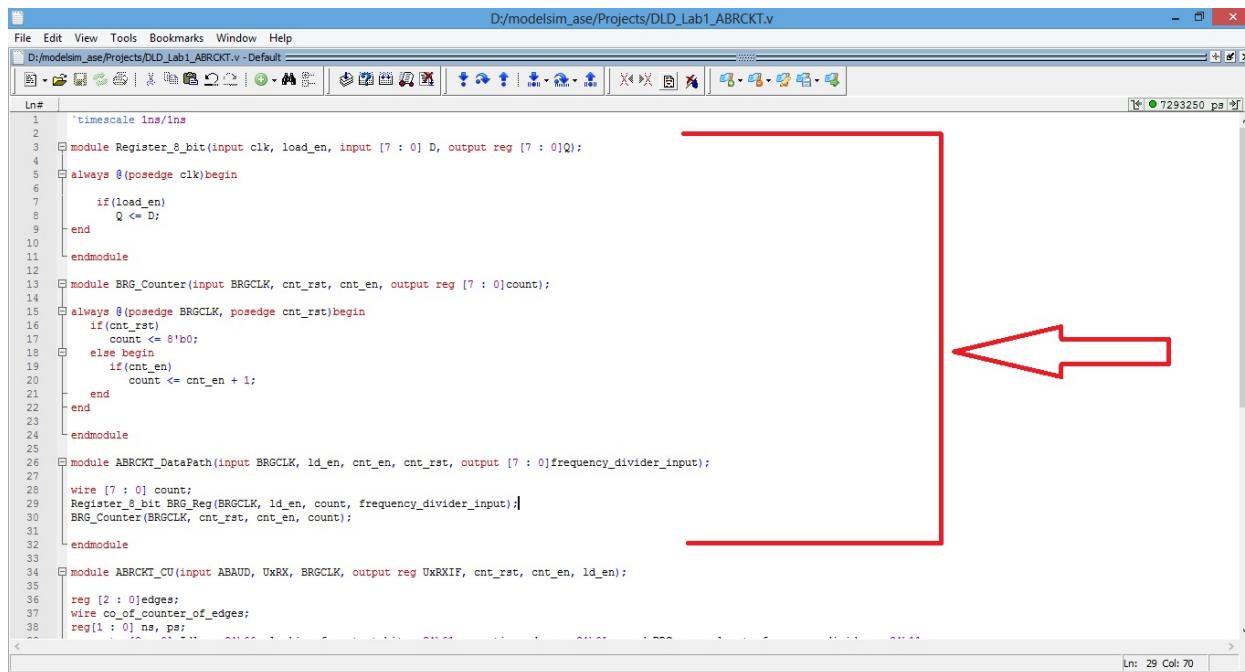
ABRCKT's Controller:



The screenshot shows the ModelSim simulation software interface with the file D:/modelsim_ase/Projects/DLD_Lab1_ABRCKT.v open. The code is a Verilog module for the ABRCKT controller. It includes modules for Data Path, BRG Counter, and CU. The CU module contains logic for looking for start bits, counting edges, and sending data to the frequency divider. A red arrow points from the bottom right towards the CU module code.

```
File Edit View Tools Bookmarks Window Help
D:/modelsim_ase/Projects/DLD_Lab1_ABRCKT.v - Default
Ln# 21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
717
718
719
719
720
721
722
723
724
725
726
727
727
728
729
729
730
731
732
733
734
735
736
737
737
738
739
739
740
741
742
743
744
745
745
746
747
747
748
749
749
750
751
752
753
754
755
756
756
757
758
758
759
759
760
761
762
763
764
765
765
766
767
767
768
768
769
769
770
771
772
773
774
775
775
776
777
777
778
778
779
779
780
781
782
783
784
785
785
786
787
787
788
788
789
789
790
791
792
793
794
795
795
796
797
797
798
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
817
818
819
819
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
```

ABRCKT's Data Path:



The screenshot shows the ModelSim simulation software interface with the file D:/modelsim_ase/Projects/DLD_Lab1_ABRCKT.v open. The code is a Verilog module for the ABRCKT data path. It includes modules for Register_8_bit, BRG Counter, and ABRCKT_DataPath. The BRG Counter module contains logic for counting edges and sending data to the frequency divider. A red arrow points from the bottom right towards the BRG Counter code.

```
File Edit View Tools Bookmarks Window Help
D:/modelsim_ase/Projects/DLD_Lab1_ABRCKT.v - Default
Ln# 1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
378
379
380
381
382
383
384
385
386
387
388
388
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
418
419
420
421
422
423
424
425
426
427
428
428
429
430
431
432
433
434
435
436
437
438
438
439
440
441
442
443
444
445
446
447
447
448
449
449
450
451
452
453
454
455
456
457
457
458
459
459
460
461
462
463
464
465
466
466
467
468
468
469
469
470
471
472
473
474
475
476
477
477
478
479
479
480
481
482
483
484
485
486
487
487
488
489
489
490
491
492
493
494
495
496
497
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
517
518
519
519
520
521
522
523
524
525
526
527
527
528
529
529
530
531
532
533
534
535
536
537
537
538
539
539
540
541
542
543
544
545
546
546
547
548
548
549
549
550
551
552
553
554
555
556
557
557
558
559
559
560
561
562
563
564
565
566
566
567
568
568
569
569
570
571
572
573
574
575
576
576
577
578
578
579
579
580
581
582
583
584
585
586
587
587
588
589
589
590
591
592
593
594
595
596
596
597
598
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
616
617
618
618
619
619
620
621
622
623
624
625
626
627
627
628
629
629
630
631
632
633
634
635
636
637
637
638
639
639
640
641
642
643
644
645
646
646
647
648
648
649
649
650
651
652
653
654
655
656
657
657
658
659
659
660
661
662
663
664
665
666
666
667
668
668
669
669
670
671
672
673
674
675
676
676
677
678
678
679
679
680
681
682
683
684
685
686
686
687
688
688
689
689
690
691
692
693
694
695
695
696
697
697
698
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
716
717
718
718
719
719
720
721
722
723
724
725
726
727
727
728
729
729
730
731
732
733
734
735
736
736
737
738
738
739
739
740
741
742
743
744
745
745
746
747
747
748
748
749
749
750
751
752
753
754
755
756
756
757
758
758
759
759
760
761
762
763
764
765
765
766
767
767
768
768
769
769
770
771
772
773
774
775
775
776
777
777
778
778
779
779
780
781
782
783
784
785
785
786
787
787
788
788
789
789
790
791
792
793
794
794
795
795
796
796
797
797
798
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
815
816
817
817
818
818
819
819
820
821
822
823
824
825
825
826
827
827
828
828
829
829
830
831
832
833
834
835
835
836
837
837
838
838
839
839
840
841
842
843
844
844
845
846
846
847
847
848
848
849
849
850
851
852
853
854
855
855
856
857
857
858
858
859
859
860
861
862
863
864
864
865
866
866
867
867
868
868
869
869
870
871
872
873
874
874
875
875
876
876
877
877
878
878
879
879
880
881
882
883
884
884
885
885
886
886
887
887
888
888
889
889
890
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
901
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
```

ABRCKT's Top level:



```

D:/modelsim_ase/Projects/DLD_Lab1_ABRCKT.v
File Edit View Tools Bookmarks Window Help
D:/modelsim_ase/Projects/DLD_Lab1_ABRCKT.v - Default
Ln# 32
32 endmodule
33
34 module ABRCKT_CU(input ABAUD, UxRX, BRGCLK, output reg UxRXIF, cnt_rst, cnt_en, ld_en);
35
36 reg [2 : 0]edges;
37 wire co_of_counter_of_edges;
38 reg[1 : 0] ns, ps;
39 parameter [2 : 0] Idle = 2'b00, looking_for_start_bit = 2'b01, counting_edges = 2'b10, send_BRG_req_value_to_frequency_divider = 2'b11;
40 always @ (ABAUD, UxRX, ps, co_of_counter_of_edges) begin
41 case(ps)
42   Idle: begin ns = ABAUD ? looking_for_start_bit : Idle; ld_en = 0; UxRXIF = 0; edges = 3'b00; end
43   looking_for_start_bit: begin ns = UxRX ? looking_for_start_bit : counting_edges; cnt_rst = 1; end
44   counting_edges: begin ns = co_of_counter_of_edges ? send_BRG_req_value_to_frequency_divider : counting_edges; cnt_en = 1; cnt_rst = 0; edges = UxRX ? edges + 1 : end
45   send_BRG_req_value_to_frequency_divider: begin ns = Idle; ld_en = 1; UxRXIF = 1; end
46   default: begin ns = Idle; end
47 endcase
48 end
49
50 always @(posedge BRGCLK) begin
51   ps <= ns;
52 end
53
54 assign co_of_counter_of_edges = (edges >= 5) ? 1 : 0;
55
56 endmodule
57
58 module ABRCKT(input ABAUD, UxRX, BRGCLK, output [7 : 0]frequency_divider_input, output UxRXIF);
59
60 wire cnt_rst, cnt_en, ld_en;
61 ABRCKT_DataPath Data_Path(BRGCLK, ld_en, cnt_en, cnt_rst, frequency_divider_input);
62 ABRCKT_CU CU(ABAUD, UxRX, BRGCLK, UxRXIF, cnt_rst, cnt_en, ld_en);
63
64 endmodule
65
66
67
68
69

```

Schematic symbol of Quartus's synthesized ABRCKT:

