



Trinity College Dublin

Coláiste na Tríonóide, Baile Átha Cliath

The University of Dublin

ASSIGNMENT-2

DIGITAL SYSTEM DESIGN

SUBMITTING TO: SHREEJITH SHANKER

SUBMITTED BY: SHRUTI KATHURIA

Aim:

The purpose of this session is to bring together elements from all your Labs, but particularly Lab F-G, and give you experience with developing and testing a larger sequential design and targeting it to the FPGA board. This assignment is worth 25% of 3C7.

Sequence Detector

You need to design a system that comprises an LFSR, a sequence detecting Finite State Machine (FSM), and a counter, in order to count the number of times a certain codeword (10, 12 bits long) is issued in the stream of bits generated by the LFSR in a full cycle of that LFSR. Each student will be given a different sequence/codeword, found in a separate handout LFSR_code_setup.pdf.

DATA USED IN THE ASSIGNMENT

TCD ID	NAME	BOARD NUMBER	LFSR_WIDTH	SEED VALUE	PATTERN
21355061	KATHURIA SHRUTI	Basys3108	24 xor	100111111001000000000000	11001010000

WORKING OF THE DESIGN:

At first, we made different modules

•Clock:

This helps us to control the speed of the BASYS 3 board. The output of the clock is send to the LFSR module from which we obtain a 24 bit number and max tick indicating that there are $2^{24}-1$ cycles.

•State Machine Module:

In this we drew the Moore diagram of the state 11001010000 and then minimized the state table. Inside the state machine I gave 4-bit long variables to store my state as 11 bits were available and that can be stored in 4 bits. Used a basic switch case for determining the present and next state.

•Counter:

Now this 1 bit found was provided to the counter which also worked on the clock edge, if the found was high the bcd counter increased its count by 1 until it is either cleared by reset or max tick or crosses the value of 9999.

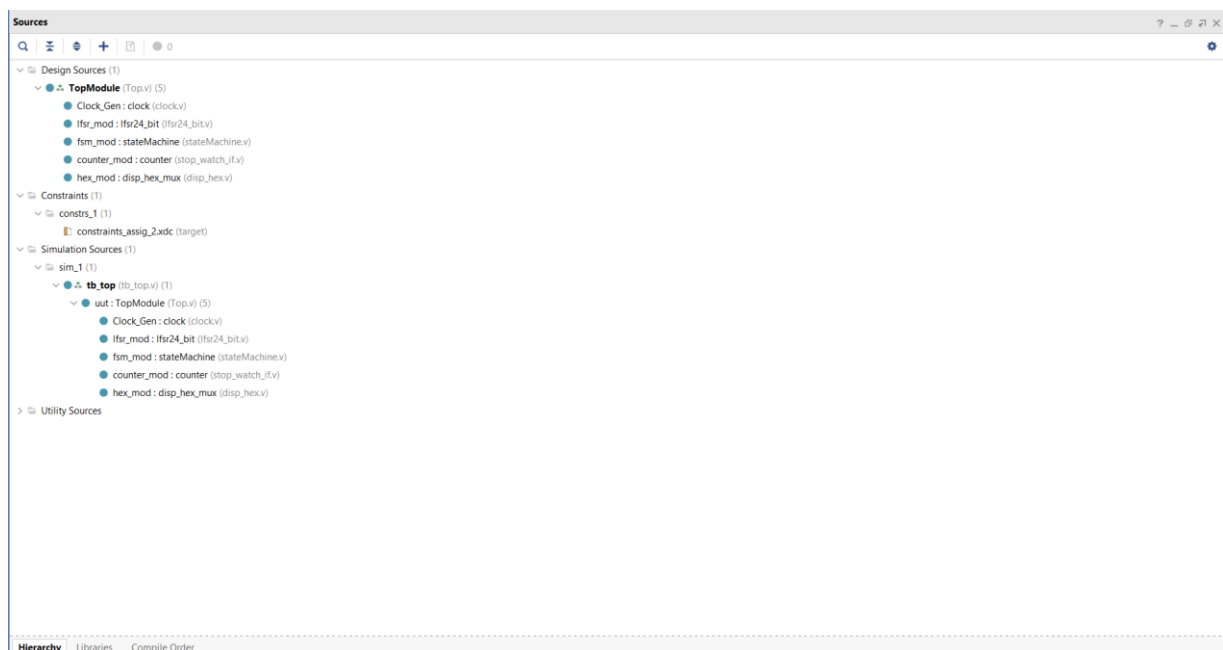
•Seven Segment Display:

We had to display the count on the seven-segment display of thebasys-3 board. Using the provide code for displaying hex from Lab G. The only thing that had to be taken care of was that the clock provide must be the boards one as we need high speed to execute the multiplexing on the board.

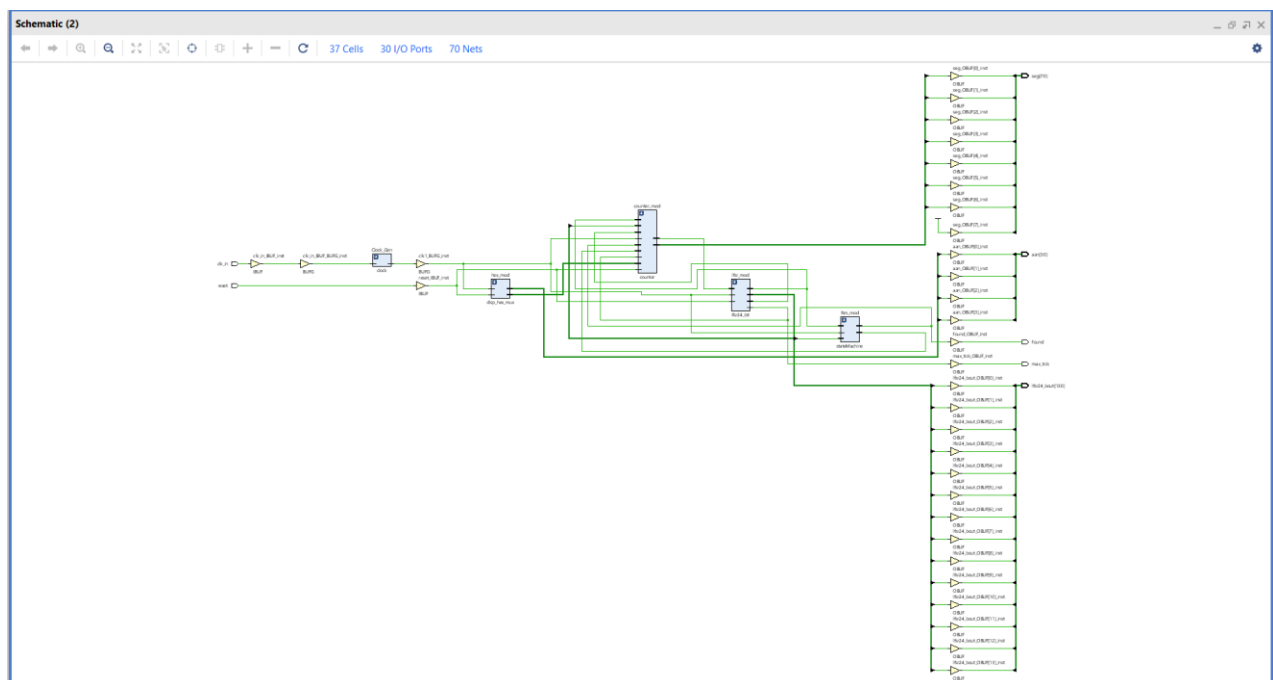
While Designing the Finite State Machine the overlapping is considered.

LFSR:

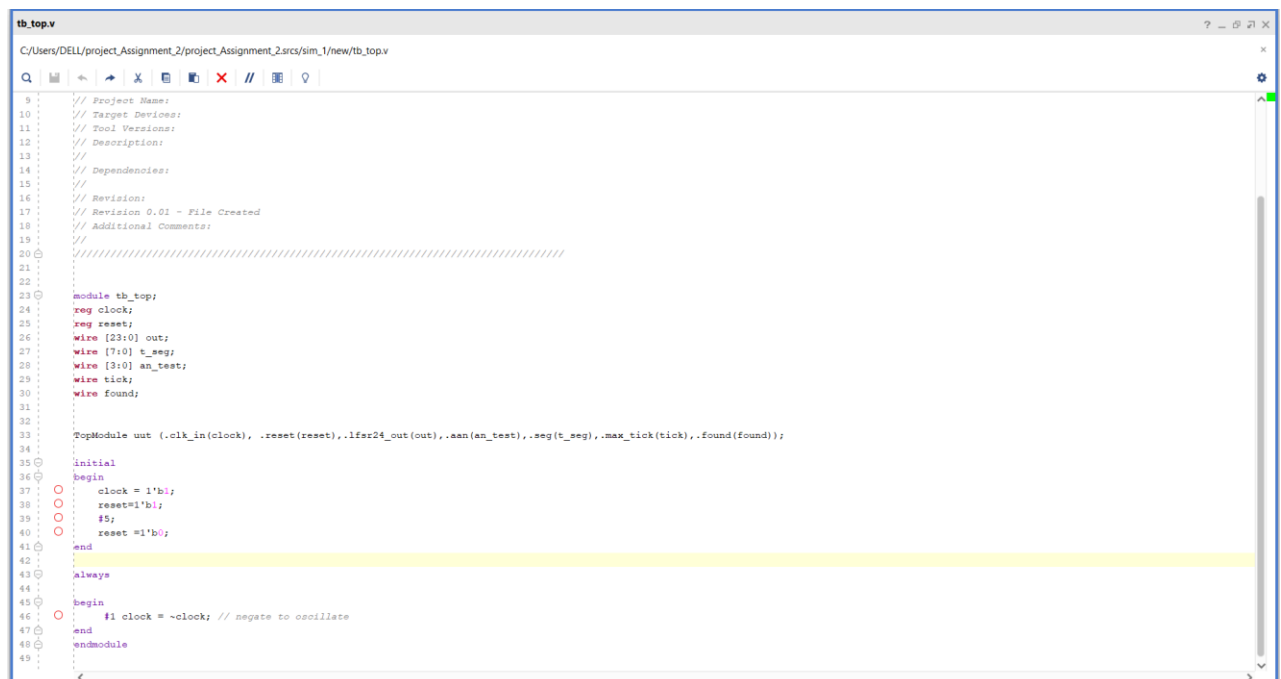
The seed value is given according to which the LFSR code is written and the tap values are taken from the XILINUX table. We should get the seed value when the reset is high in LFSR.



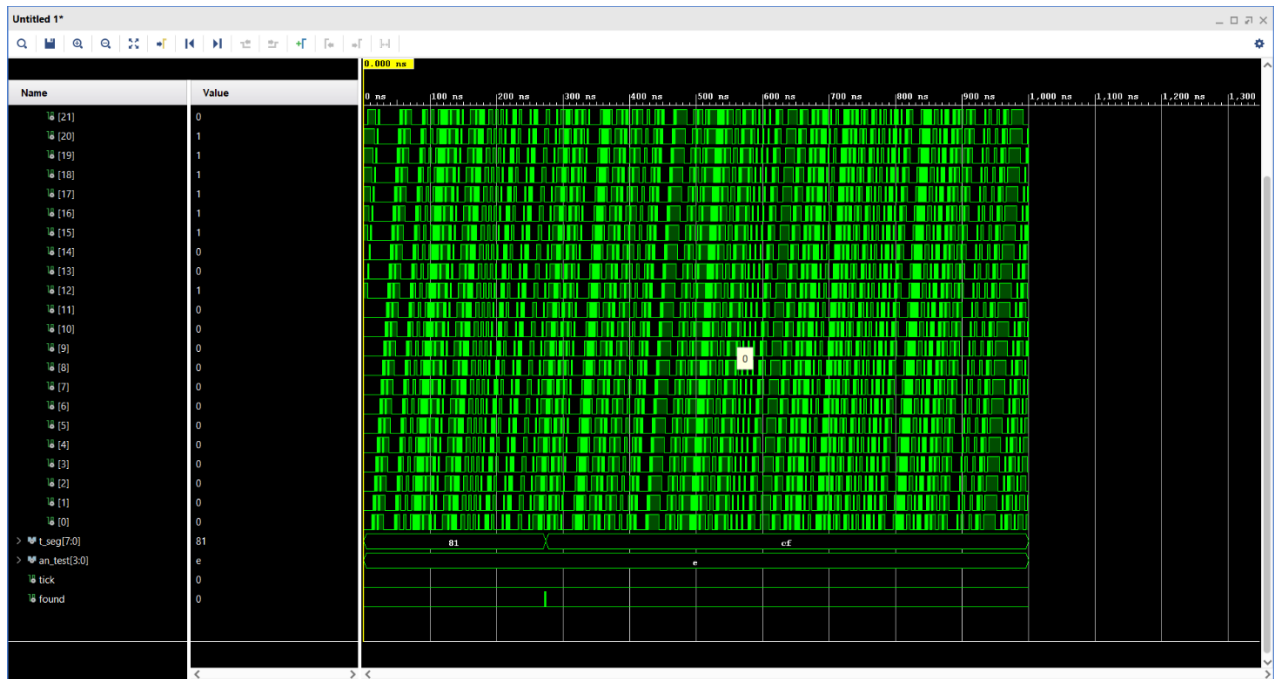
Hierarchy

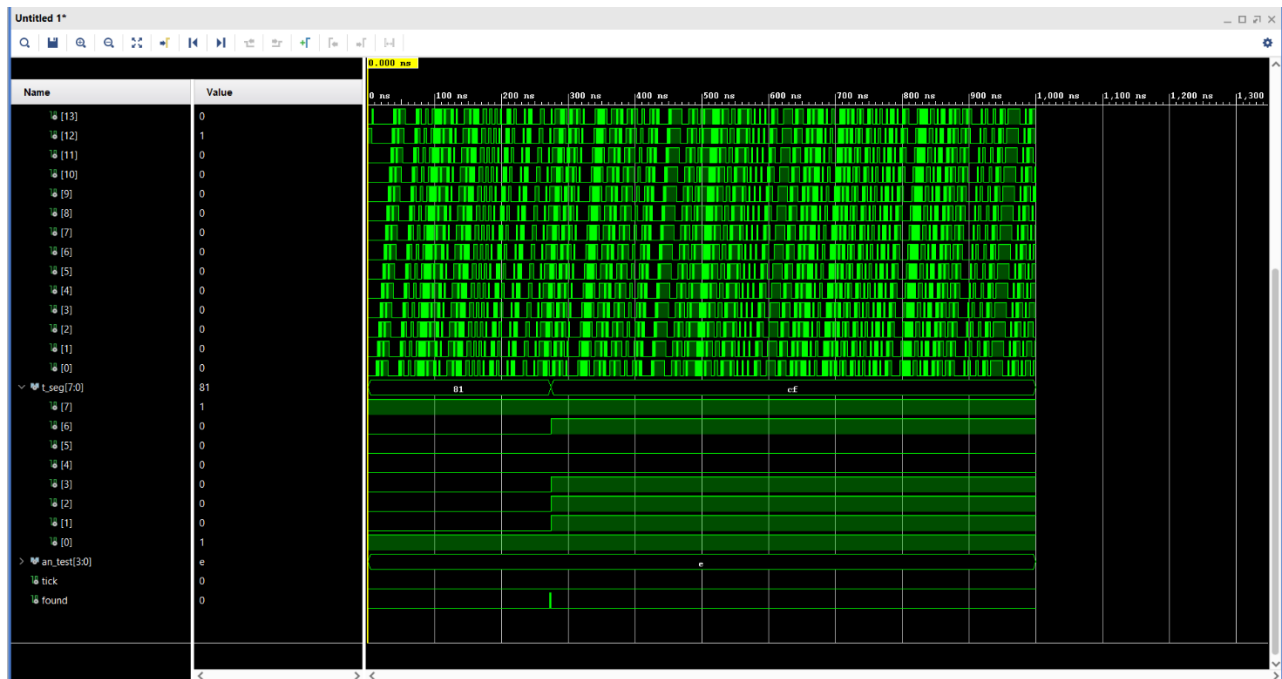
SCHEMATIC DIAGRAM:**TESTBENCH:**

Reset is set to high



When we use this testbench the output is:





We can see whenever the found is high, on the next positive clock edge the counter increase the count.

FINDING THE PATTERN ACCORDING TO MOORE:

As there are 11 bits we take 12 bits to make the state diagram. Among which state A and L are same so it gets minimised.

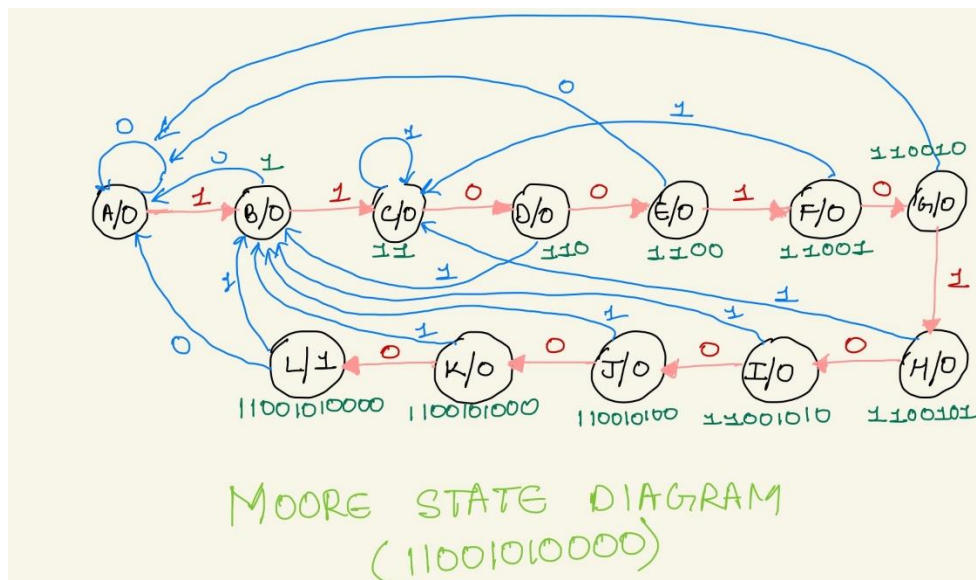
Current State	Next State		Output	
	Input 0	Input 1	Input 0	Input 1
A	A	B	0	0
B	A	C	0	0
C	D	C	0	0
D	E	B	0	0
E	A	F	0	0
F	G	A	0	0
G	A	H	0	0
H	I	C	0	0
I	J	B	0	0
J	K	B	0	0
K	L	B	1	0

B	B,C									
C	A,D B,C	A,D								
D	A,E	A,E C,B	D,E C,B							
E	B,F	C,F	A,D C,F	A,E B,F						
F	A,G A,B	A,G A,C	D,G A,C	E,G A,B	A,G A,F					
G	B,H	A,H	A,D C,H	A,E B,H	F,H	A,G A,H				
H	A,I B,C	A,I	D,I	E,I B,C	A,I C,F	G,I A,C	A,I C,H			
I	A,J	A,J C,B	D,J C,B	E,J	A,J B,F	G,J A,B	A,G H,B	I,J C,B		
J	A,K	A,K C,B	D,K C,B	E,K	A,K B,F	G,K A,B	A,K B,H	I,K C,B	J,K	
K	A,L	A,L C,B	D,L C,B	E,L	A,L B,F	G,L A,B	A,L B,H	I,L C,B	J,L	K,L
L	✓	C,B	D,B C,B	A,E	F,B	A,G A,B	H,B	I,B C,B	A,J	A,K
	A	B	C	D	E	F	G	H	I	J

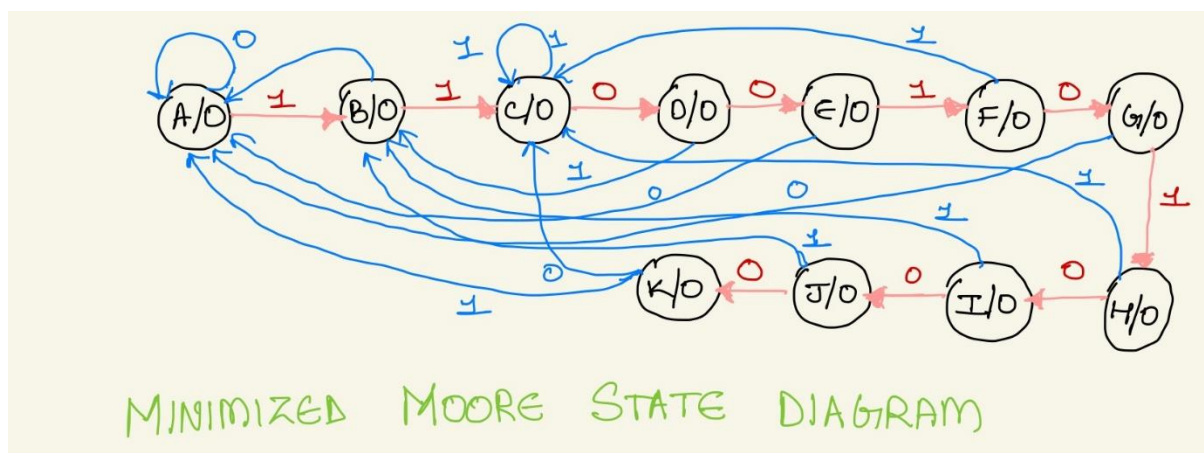
B	B,C									
C	A,D B,C	A,D								
D	A,E	A,E C,B	D,E C,B							
E	B,F	C,F	A,D C,F	A,E B,F						
F	A,G A,B	A,G A,C	D,G A,C	E,G A,B	A,G A,F					
G	B,H	A,H	A,D C,H	A,E B,H	F,H	A,G A,H				
H	A,I B,C	A,I	D,I	E,I B,C	A,I C,F	G,I A,C	A,I C,H			
I	A,J	A,J C,B	D,J C,B	E,J	A,J B,F	G,J A,B	A,G H,B	I,J C,B		
J	A,K	A,K C,B	D,K C,B	E,K	A,K B,F	G,K A,B	A,K B,H	I,K C,B	J,K	
K	A,L	A,L C,B	D,L C,B	E,L	A,L B,F	G,L A,B	A,L B,H	I,L C,B	J,L	K,L
L	✓	C,B	D,B C,B	A,E	F,B	A,G A,B	H,B	I,B C,B	A,J	A,K
	A	B	C	D	E	F	G	H	I	J

MINIMISING TABLE

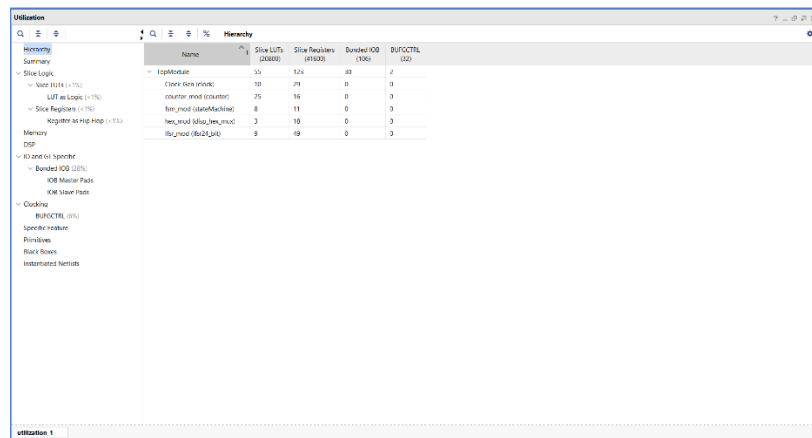
FINITE STATE DIAGRAM



MINIMIZED STATE DIAGRAM:

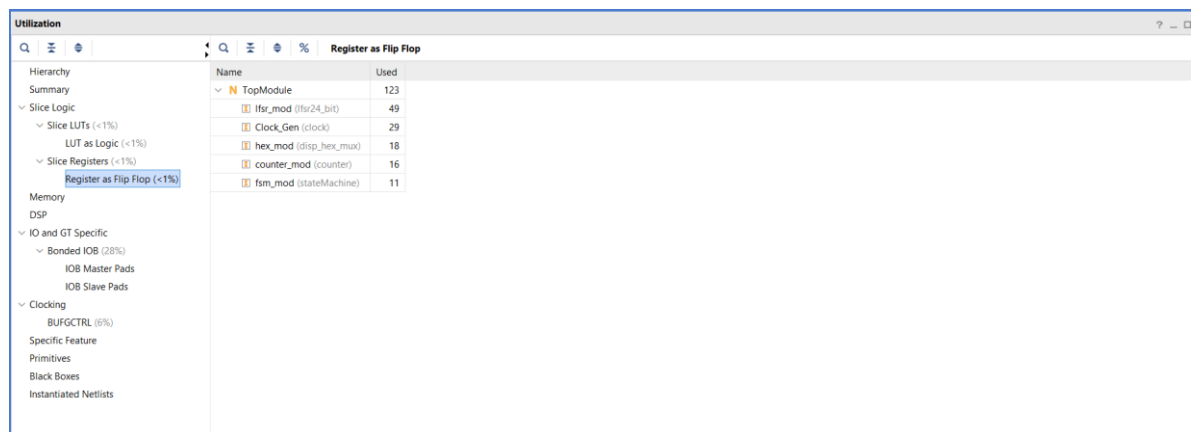


UTILIZATION REPORT:



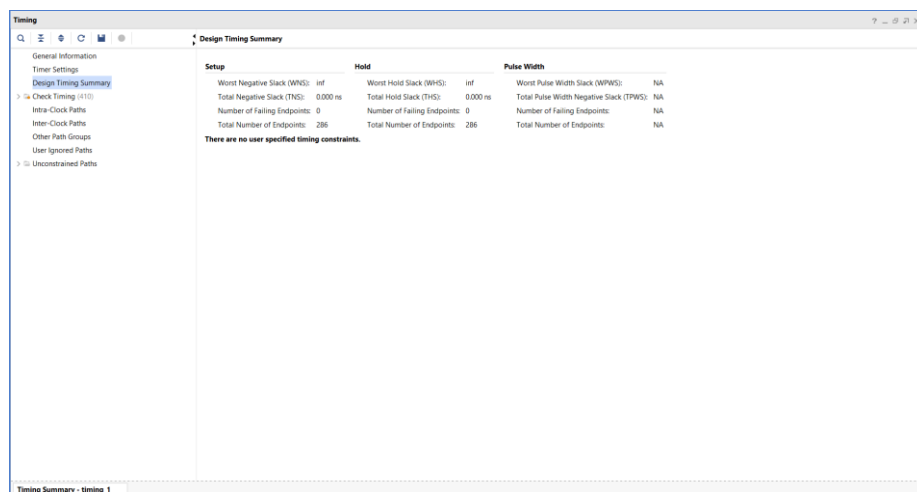
Hierarchy	Name	Slice LUTs (00000)	Slice Registers (01000)	Bonded IOB (100)	BUFGCTRL (10)
Summary					
▼ Slice Logic		55	124	89	2
▼ Slice LUTs (<1%)					
▼ LUT as Logic (<1%)					
▼ Slice Registers (<1%)					
▼ Register as Flip Flop (<1%)					
Memory					
DSP					
▼ IO and GT Specific					
▼ Bonded IOB (28%)					
IOB Master Pads					
IOB Slave Pads					
▼ Clocking					
BUFGCTRL (6%)					
Specific Feature					
Primitives					
Black Boxes					
Instantiated Netlists					

UTILIZATION REPORT ACCORDING TO REGISTER FLIP FLOPS:



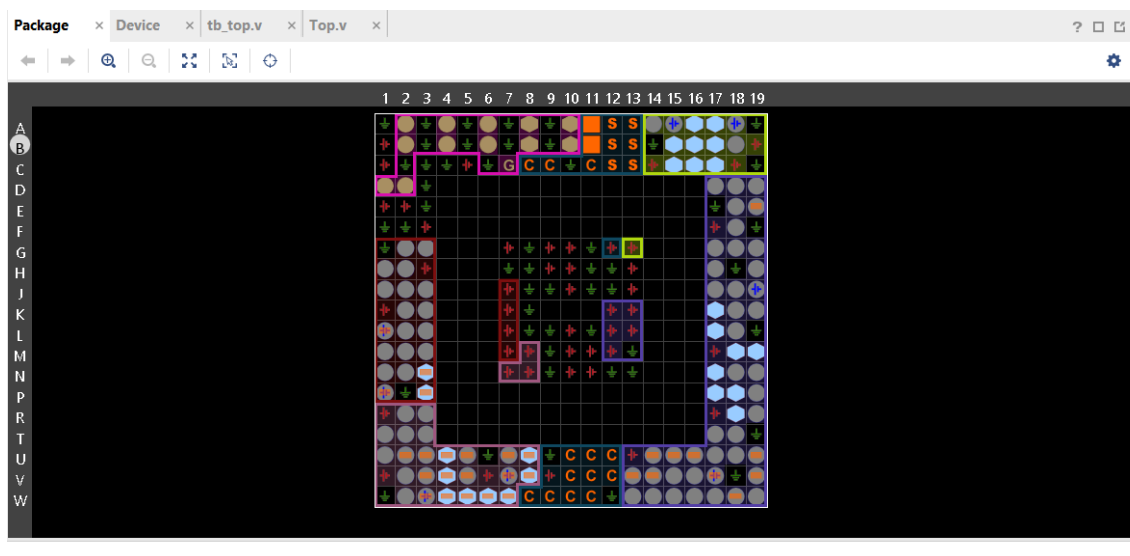
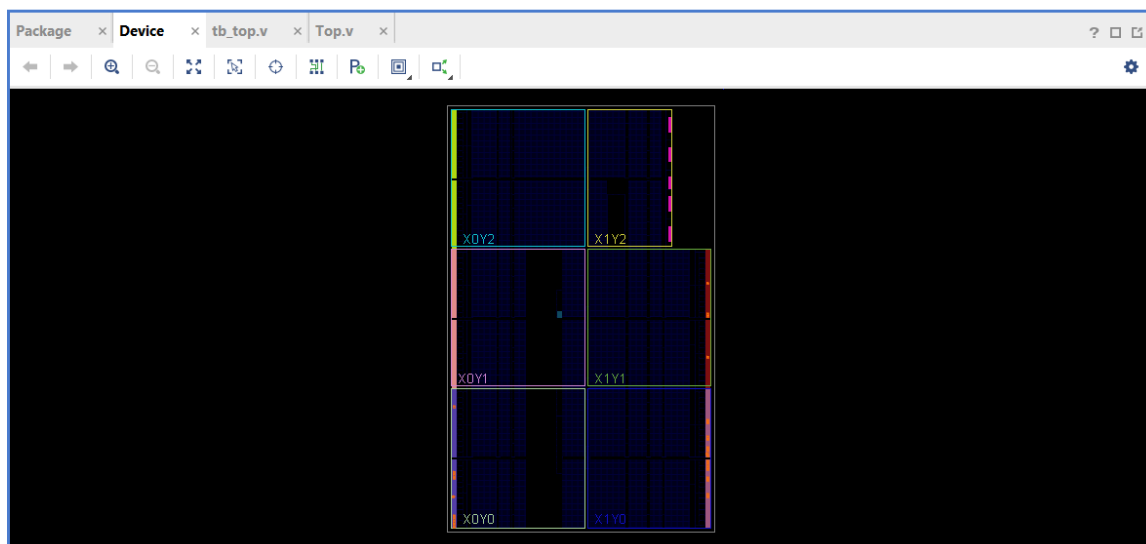
Hierarchy	Name	Used
Summary		
▼ Slice Logic		
▼ Slice LUTs (<1%)		
▼ LUT as Logic (<1%)		
▼ Slice Registers (<1%)		
▼ Register as Flip Flop (<1%)		
Memory		
DSP		
▼ IO and GT Specific		
▼ Bonded IOB (28%)		
IOB Master Pads		
IOB Slave Pads		
▼ Clocking		
BUFGCTRL (6%)		
Specific Feature		
Primitives		
Black Boxes		
Instantiated Netlists		

TIMING REPORT:



Setup	Hold	Pulse Width
Worst Negative Slack (WNS): inf	Worst Hold Slack (WHS): inf	Worst Pulse Width Slack (WPWS): NA
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): NA
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: NA
Total Number of Endpoints: 286	Total Number of Endpoints: 286	Total Number of Endpoints: NA

There are no user specified timing constraints.

PACKAGE:**DEVICE:**

1)CLOCK.V

2)LFSR24_BIT.V

3) COUNTER.V

```
stop_watch.v
```

```
1 // Listing 4.18
2 module counter
3 (
4     input wire clk,
5     input wire ps, clr,
6     output wire [3:0] d3,d2, d1, d0
7 );
8
9 // declaration
10 localparam DSIZE = 10000000;
11 reg [3:0] ma_reg;
12 wire [23:0] ma_next;
13 reg [3:0] d3_reg,d2_reg, d1_reg, d0_reg;
14 reg [3:0] d3_next,d2_next, d1_next, d0_next;
15 wire ma_tick;
16
17 // body
18 // register
19 always @(posedge clk)
20 begin
21     ma_reg <= ma_next;
22     d3_reg <= d3_next;
23     d2_reg <= d2_next;
24     d1_reg <= d1_next;
25     d0_reg <= d0_next;
26 end
27
28 // next-state logic
29 // 3.1 new tick generator: mod-8000000
30 assign ma_next = (clk && (ma_reg==DSIZE-4) go) ? 4'b0 :
31                 (go) ? ma_reg + 1 :
32                 ma_reg;
33 assign ma_tick = (ma_reg==DSIZE) ? 1'b1 : 1'b0;
34 // 3.2 tick and counter
35 always @*
36 begin
37     // default: keep the previous value
38     d0_next = d0_reg;
39     d1_next = d1_reg;
40     d2_next = d2_reg;
41     d3_next = d3_reg;
42 end
```

```

Step watch #1x
C:\Users\GOLL\project_Assigner_2\project_Assigner_2\source\1\reports\Provided stopwatch codepath watch_1x\
Q 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 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1
```

```

disp_hex.v
C:\Users\DELL\project_Assignment_2\srcs\source_1\hew\disp_hex.v

1 // timescale line | ipse
2 module disp_hex_mux
3 (
4     input wire clk, reset,
5     input wire [3:0] hex0, hex2, hex1, hex0, // hex digits
6     input wire [3:0] dp_in, // 4 decimal points
7     output reg [7:0] an, // enable 2-out-of-4 asserted low
8     output reg [7:0] cseg // 2nd segments
9 );
10 // constant declaration
11 // refreshing rate around 800 Hz (50 MHz/2^16)
12 localparam M = 18;
13 // internal signal declaration
14 reg [M-1:0] q_seg;
15 wire [M-1:0] q_next;
16 reg [3:0] hex_in;
17 reg dp;
18 // 8-bit counter
19 // register
20 always @(posedge clk)
21 if (reset)
22     q_seg <= 0;
23 else
24     q_seg <= q_next;
25 // next-state logic
26 assign q_next = q_seg + 1;
27 // 2 MHz of counter to control 4-to-1 multiplexing
28 // and to generate active-low enable signal
29 always #*
30 case (q_seg[M-1:M-2])
31     2'b01:
32         begin
33             an = 4'b1100;
34             hex_in = hex0;
35             dp = dp_in[0];
36         end
37     2'b01:
38         begin
39             an = 4'b1010;
40             hex_in = hex1;
41             dp = dp_in[1];
42         end
43     // ...
44 endcase

```

```

disp_hex.v
C:/Users/DELL/project_Assignment_2/project_Assignment_2/srcs/sources_1/new/disp_hex.v

37: 2'b01;
38: begin
39:   an = 4'b1001;
40:   hex_in = hex1;
41:   dp = dp_in[1];
42: end
43: 2'b01;
44: begin
45:   an = 4'b0111;
46:   hex_in = hex2;
47:   dp = dp_in[2];
48: end
49: 2'b11;
50: begin
51:   an = 4'b0111;
52:   hex_in = hex3;
53:   dp = dp_in[3];
54: end
55: default:
56:   begin
57:     an = 4'b0111;
58:     hex_in = hex3;
59:     dp = dp_in[3];
60:   end
61: endcase
62: // hex to seven-segment led display
63:
64: always @*
65: begin
66:   case(hex_in)
67:     4'h1: seg[6:0] = 7'b0000001;
68:     4'h1: seg[6:0] = 7'b0001111;
69:     4'h1: seg[6:0] = 7'b0000011;
70:     4'h1: seg[6:0] = 7'b0000111;
71:     4'h1: seg[6:0] = 7'b0011001;
72:     4'h1: seg[6:0] = 7'b0100101;
73:     4'h1: seg[6:0] = 7'b0100001;
74:     4'h1: seg[6:0] = 7'b0011111;
75:     4'h1: seg[6:0] = 7'b0000001;
76:     4'h1: seg[6:0] = 7'b0000101;
77:     4'h1: seg[6:0] = 7'b0001001;
78:   end
79:   endcase
80:   // an = seg[6:0]
81:   end

```

```

disp_hex.v
C:/Users/DELL/project_Assignment_2/project_Assignment_2/srcs/sources_1/new/disp_hex.v

49: end
50: 2'b11;
51: begin
52:   an = 4'b0111;
53:   hex_in = hex3;
54:   dp = dp_in[3];
55: end
56: default:
57:   begin
58:     an = 4'b0111;
59:     hex_in = hex3;
60:     dp = dp_in[3];
61:   end
62: endcase
63: // hex to seven-segment led display
64:
65: always @*
66: begin
67:   case(hex_in)
68:     4'h1: seg[6:0] = 7'b0000001;
69:     4'h1: seg[6:0] = 7'b0001111;
70:     4'h1: seg[6:0] = 7'b0000011;
71:     4'h1: seg[6:0] = 7'b0000111;
72:     4'h1: seg[6:0] = 7'b0011001;
73:     4'h1: seg[6:0] = 7'b0100101;
74:     4'h1: seg[6:0] = 7'b0100001;
75:     4'h1: seg[6:0] = 7'b0011111;
76:     4'h1: seg[6:0] = 7'b0000001;
77:     4'h1: seg[6:0] = 7'b0000101;
78:     4'h1: seg[6:0] = 7'b0001001;
79:     4'h1: seg[6:0] = 7'b0000011;
80:     4'h1: seg[6:0] = 7'b0000111;
81:     4'h1: seg[6:0] = 7'b0011001;
82:   end
83:   default:
84:     seg[6:0] = 7'b0111001;
85:   endcase
86:   seg[7] = dp;
87: end
88: endmodule
89:

```

5) STATE MACHINE

```

stateMachine.v
C:/Users/DELL/project_Assignment_2/project_Assignment_2/srcs/sources_1/new/stateMachine.v

1: timescale 1ns / 1ps
2: timescale 1ns / 1ps
3: module stateMachine
4: (
5:   input wire clk, reset,
6:   input wire seg,
7:   output reg tick
8: );
9: // symbolic state declaration
10: localparam A = 4'b0001;
11: A = 4'b0001;
12: D = 4'b0011;
13: D = 4'b0011;
14: E = 4'b0101;
15: F = 4'b0101;
16: G = 4'b0110;
17: H = 4'b0111;
18: I = 4'b1000;
19: J = 4'b1001;
20: K = 4'b1001;
21:
22: reg [3:0] state_reg, state_next;
23:
24: always @(posedge clk, posedge reset)
25: begin
26:   if (reset)
27:     state_reg <= A;
28:   else
29:     state_reg <= state_next;
30: end
31: // next-state logic and output logic
32: always @*
33: begin
34:   state_next = state_reg;
35:   // default: state is same
36:   tick = 1'b0; // default: outputs: 0
37:   case (state_reg)
38:     A:
39:       if (seg)
40:         state_next = H;
41:       else
42:         state_next = A;
43:   end
44: end

```

```
stateMachine.v
C:/Users/DELL/project_Assignment_2/project_Assignment_2/srcs/sources_1/new/stateMachine.v

37:0 case (state_reg)
38:0 A:
39:0 if (enq)
40:0 state_next = B;
41:0 else
42:0 state_next = A;
43:0 B:
44:0 if (enq)
45:0 state_next = C;
46:0 else
47:0 state_next = A;
48:0 C:
49:0 if (~enq)
50:0 state_next = D;
51:0 else
52:0 state_next = C;
53:0 D:
54:0 if (~enq)
55:0 state_next = E;
56:0 else
57:0 state_next = B;
58:0 E:
59:0 if (enq)
60:0 state_next = F;
61:0 else
62:0 state_next = C;
63:0 F:
64:0 if (~enq)
65:0 state_next = D;
66:0 else
67:0 state_next = A;
68:0 G:
69:0 if (enq)
70:0 state_next = H;
71:0 else
72:0 state_next = C;
73:0 H:
74:0 if (~enq)
75:0 state_next = I;
76:0 else
77:0 state_next = B;
```

```
stateMachine.v
C:/Users/DELL/project_Assignment_2/project_Assignment_2/srcs/sources_1/new/stateMachine.v

68:0 state_next = G;
69:0 else
70:0 state_next = A;
71:0 G:
72:0 if (enq)
73:0 state_next = B;
74:0 else
75:0 state_next = C;
76:0 H:
77:0 if (~enq)
78:0 state_next = D;
79:0 else
80:0 state_next = J;
81:0 J:
82:0 if (~enq)
83:0 state_next = K;
84:0 else
85:0 state_next = B;
86:0 K:
87:0 if (~enq)
88:0 begin
89:0 state_next = A;
90:0 lock1'b1;
91:0 end
92:0 else
93:0 state_next = B;
94:0 L:
95:0 default:
96:0 begin
97:0 state_next = A;
98:0 lock1'b1;
99:0 end
100:0 endcase
101:0 end
102:0 endmodule
103:0 endmodule
104:0 endmodule
105:0
```

CONSTRAINTS (XDC FILE)

```
constraints_assig.2.xdc
C:/Users/DELL/project_Assignment_2/project_Assignment_2/srcs/constrs_1/new/constraints_assig.2.xdc

1: set_property TOPSTANDARD LVCMOS33 [get_ports {aen[0]}]
2: set_property TOPSTANDARD LVCMOS33 [get_ports {aen[1]}]
3: set_property TOPSTANDARD LVCMOS33 [get_ports {aen[1]}]
4: set_property TOPSTANDARD LVCMOS33 [get_ports {aen[0]}]
5: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[13]}]
6: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[12]}]
7: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[11]}]
8: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[10]}]
9: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[9]}]
10: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[8]}]
11: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[7]}]
12: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[6]}]
13: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[5]}]
14: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[4]}]
15: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[3]}]
16: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[2]}]
17: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[1]}]
18: set_property TOPSTANDARD LVCMOS33 [get_ports {ifaz24_bout[0]}]
19: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[7]}]
20: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[6]}]
21: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[5]}]
22: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[4]}]
23: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[3]}]
24: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[2]}]
25: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[1]}]
26: set_property TOPSTANDARD LVCMOS33 [get_ports {enq[0]}]
27: set_property PACKAGE_PIN W4 [get_ports {aen[0]}]
28: set_property PACKAGE_PIN V4 [get_ports {aen[1]}]
29: set_property PACKAGE_PIN U4 [get_ports {aen[1]}]
30: set_property PACKAGE_PIN U2 [get_ports {aen[0]}]
31: set_property PACKAGE_PIN N3 [get_ports {ifaz24_bout[13]}]
32: set_property PACKAGE_PIN N3 [get_ports {ifaz24_bout[12]}]
33: set_property PACKAGE_PIN O3 [get_ports {ifaz24_bout[11]}]
34: set_property PACKAGE_PIN W3 [get_ports {ifaz24_bout[10]}]
35: set_property PACKAGE_PIN V3 [get_ports {ifaz24_bout[9]}]
36: set_property PACKAGE_PIN V13 [get_ports {ifaz24_bout[8]}]
37: set_property PACKAGE_PIN V14 [get_ports {ifaz24_bout[7]}]
38: set_property PACKAGE_PIN U14 [get_ports {ifaz24_bout[6]}]
39: set_property PACKAGE_PIN U15 [get_ports {ifaz24_bout[5]}]
40: set_property PACKAGE_PIN W18 [get_ports {ifaz24_bout[4]}]
41: set_property PACKAGE_PIN V19 [get_ports {ifaz24_bout[3]}]
```

```
constraints_assign_2.xdc
C:\Users\DELL\project_Assignment_2\srcs\constr_1\new\constraints_assign_2.xdc

set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[10]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
set_property PACKAGE_PIN W4 [get_ports {aan[3]}]
set_property PACKAGE_PIN V4 [get_ports {aan[2]}]
set_property PACKAGE_PIN U4 [get_ports {aan[1]}]
set_property PACKAGE_PIN U2 [get_ports {aan[0]}]
set_property PACKAGE_PIN N3 [get_ports {ifw24_bout[13]}]
set_property PACKAGE_PIN P3 [get_ports {ifw24_bout[12]}]
set_property PACKAGE_PIN U3 [get_ports {ifw24_bout[11]}]
set_property PACKAGE_PIN W3 [get_ports {ifw24_bout[10]}]
set_property PACKAGE_PIN V3 [get_ports {ifw24_bout[9]}]
set_property PACKAGE_PIN U15 [get_ports {ifw24_bout[5]}]
set_property PACKAGE_PIN V14 [get_ports {ifw24_bout[7]}]
set_property PACKAGE_PIN U14 [get_ports {ifw24_bout[6]}]
set_property PACKAGE_PIN U15 [get_ports {ifw24_bout[5]}]
set_property PACKAGE_PIN W10 [get_ports {ifw24_bout[4]}]
set_property PACKAGE_PIN V19 [get_ports {ifw24_bout[3]}]
set_property PACKAGE_PIN U19 [get_ports {ifw24_bout[2]}]
set_property PACKAGE_PIN K19 [get_ports {ifw24_bout[1]}]
set_property PACKAGE_PIN U16 [get_ports {ifw24_bout[0]}]
set_property PACKAGE_PIN V7 [get_ports {seg[7]}]
set_property PACKAGE_PIN W7 [get_ports {seg[6]}]
set_property PACKAGE_PIN W6 [get_ports {seg[5]}]
set_property PACKAGE_PIN U8 [get_ports {seg[4]}]
set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {seg[2]}]
set_property PACKAGE_PIN V5 [get_ports {seg[1]}]
set_property PACKAGE_PIN U7 [get_ports {seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports clk_in]
set_property PACKAGE_PIN W5 [get_ports clk_in]
set_property PACKAGE_PIN V17 [get_ports reset]
set_property IOSTANDARD LVCMOS33 [get_ports reset]

set_property IOSTANDARD LVCMOS33 [get_ports found]
set_property IOSTANDARD LVCMOS33 [get_ports max_tick]
set_property PACKAGE_PIN P1 [get_ports max_tick]
set_property PACKAGE_PIN L1 [get_ports found]
```

I/O Ports

Name	Direction	Board Part Pin	Board Part Interface	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength	Slew Type	Pull Type	Off-Chip Te
▼ All ports (30)														
▼ aan (4)						<input checked="" type="checkbox"/>								
aan[3]	OUT				W4	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
aan[2]	OUT				V4	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
aan[1]	OUT				U4	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
aan[0]	OUT				U2	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
▼ ifw24_bout (14)						<input checked="" type="checkbox"/>	(Multiple)	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				N3	<input checked="" type="checkbox"/>	35	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				P3	<input checked="" type="checkbox"/>	35	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				U3	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				W3	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				V3	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				V13	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				V14	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				U14	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				U15	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				W18	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				V19	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				U19	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				E19	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
ifw24_bout...	OUT				U16	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
▼ seg (8)						<input checked="" type="checkbox"/>								
seg[7]	OUT				V7	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[6]	OUT				W7	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[5]	OUT				W6	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[4]	OUT				U8	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[3]	OUT				V8	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[2]	OUT				U5	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[1]	OUT				V5	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50
seg[0]	OUT				U7	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	▼ SLOW	▼ NONE	▼	FP_VTT_50

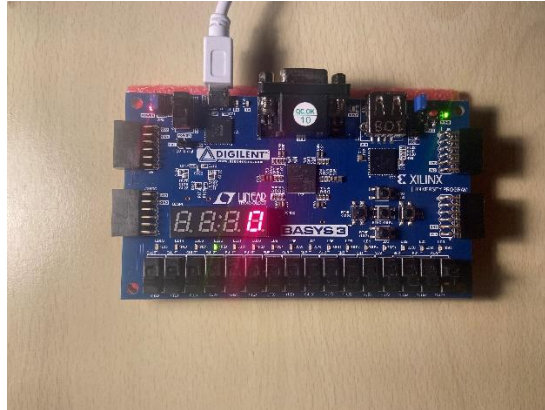
End of page (1)

I/O Ports															?		?		?	
Name	Direction	Board Part Pin	Board Part Interface	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	V _{CCO}	V _{ref}	Drive Strength	Slew Type	Pull Type	Off-Chip Termination						
▼ seg [0]																				
aan[0]	OUT				U2	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
▼ ifw24_bout [14]																				
ifw24_bout[14]	OUT					<input checked="" type="checkbox"/>	(Multiple)	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[13]	OUT				N3	<input checked="" type="checkbox"/>	35	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[12]	OUT				P3	<input checked="" type="checkbox"/>	35	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[11]	OUT				U3	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[10]	OUT				W3	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[9]	OUT				V3	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[8]	OUT				V13	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[7]	OUT				V14	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[6]	OUT				U14	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[5]	OUT				U15	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[4]	OUT				W18	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[3]	OUT				V19	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[2]	OUT				U19	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[1]	OUT				E19	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
ifw24_bout[0]	OUT				U16	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
▼ seg [1]																				
seg[7]	OUT				V7	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[6]	OUT				W7	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[5]	OUT				W6	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[4]	OUT				U8	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[3]	OUT				V8	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[2]	OUT				U5	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[1]	OUT				V5	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
seg[0]	OUT				U7	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
▼ Scalar ports [4]																				
clk_in	IN				W5	<input checked="" type="checkbox"/>	34	LVCMOS33*	3.300			NONE	NONE							
found	OUT				L1	<input checked="" type="checkbox"/>	35	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
max_tick	OUT				P1	<input checked="" type="checkbox"/>	35	LVCMOS33*	3.300	12	SLOW	NONE	NONE	FP_VTT_50						
reset	IN				V17	<input checked="" type="checkbox"/>	14	LVCMOS33*	3.300			NONE	NONE							

DEMO:

Now when we generate the bitstream and the reset is high so the output on the board is as follows:

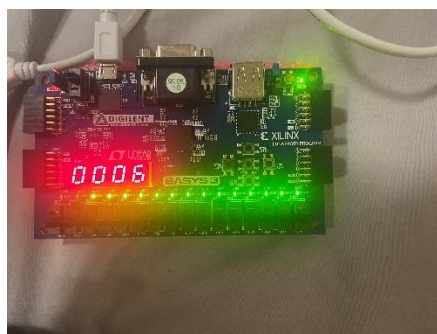
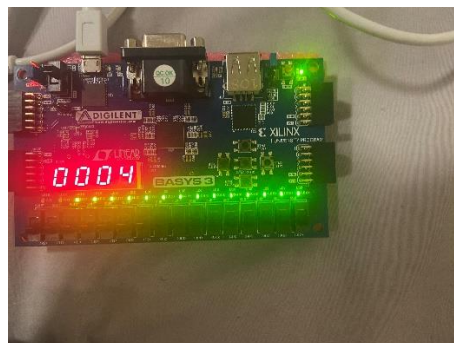
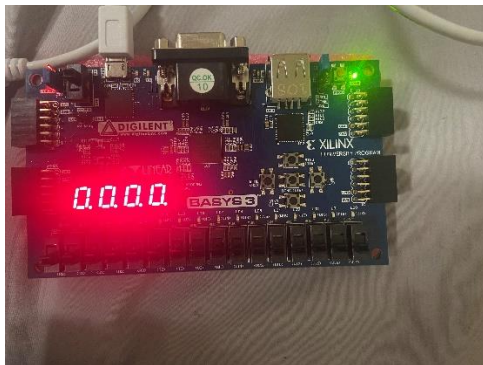
My seed value is 100111111001000000000000. I define the LFSR for 14 bits and output the seed value now comes from the left 14 bits. So now the 1 led lights up.



When we generate the full bitstream. The counter goes on and lfsr bits are also shown. The video link is attached for DEMO.

As soon as we run the bit file the led's start to blink with the speed initialised in the clock.v file.

The counter starts from 0 and goes on till 9999 on the BASYS 3 board. My LFSR cycle goes on till $2^{24}-1$. While the seven segment decoder goes on the speed of the clock is too high so the blinking of the LED's can't be seen.



The demo video is in the link below.

In the DEMO video as the max_tick becomes 1 the counter increases.

LINK: https://drive.google.com/file/d/195bqEcnWrSvQPf0e_AKP8e8wh1fiKTaG/view?usp=sharing

The clock in this project is being handled by the CLK_IN in the xdc file.

The output of the counter is the number of times the same pattern is getting generated.

-SHRUTI KATHURIA

