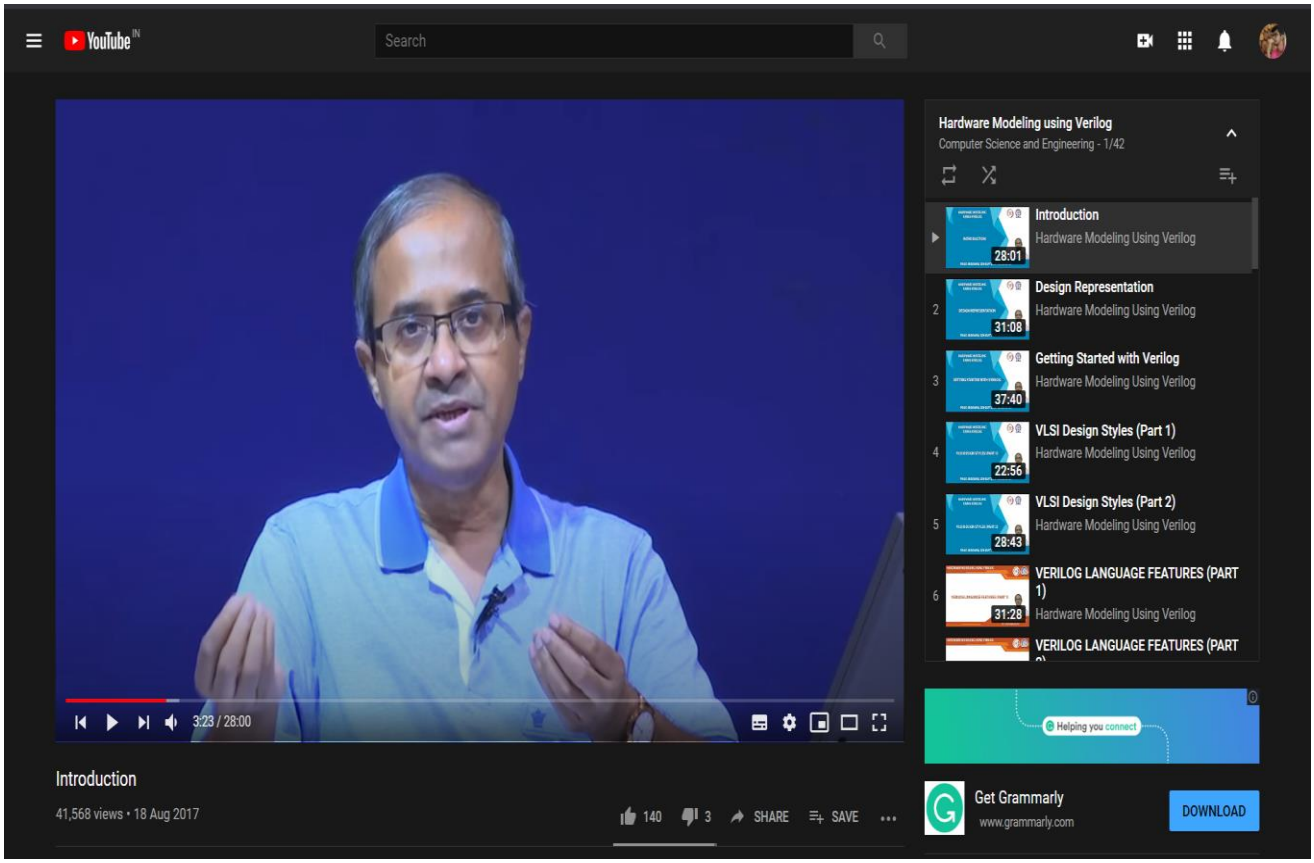
DAILY ASSESSMENT REPORT

Date:	**04 June 2020**	**Name:**	**Gagan M K**
Course:	**DIGITAL DESIGN USING HDL**	**USN:**	**4AL17EC032**
Topic:	• Hardware modelling using Verilog • FPGA and ASIC Interview questions	**Semester & Section:**	**6th sem & 'A' sec**
GitHub Repository:	**Alvas-education-foundation/Gagan-Git**		

FORENOON SESSION DETAILS

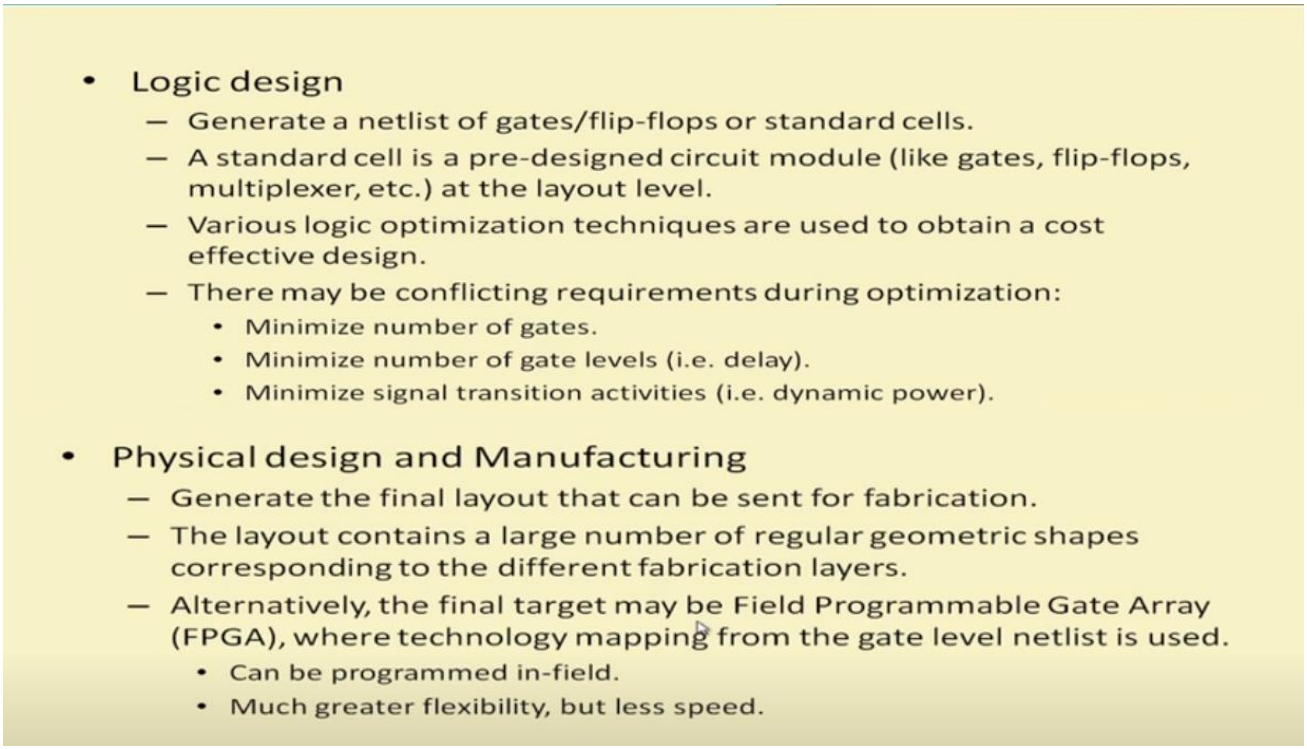
Image of session



Report – Report can be typed or hand written for up to two pages.

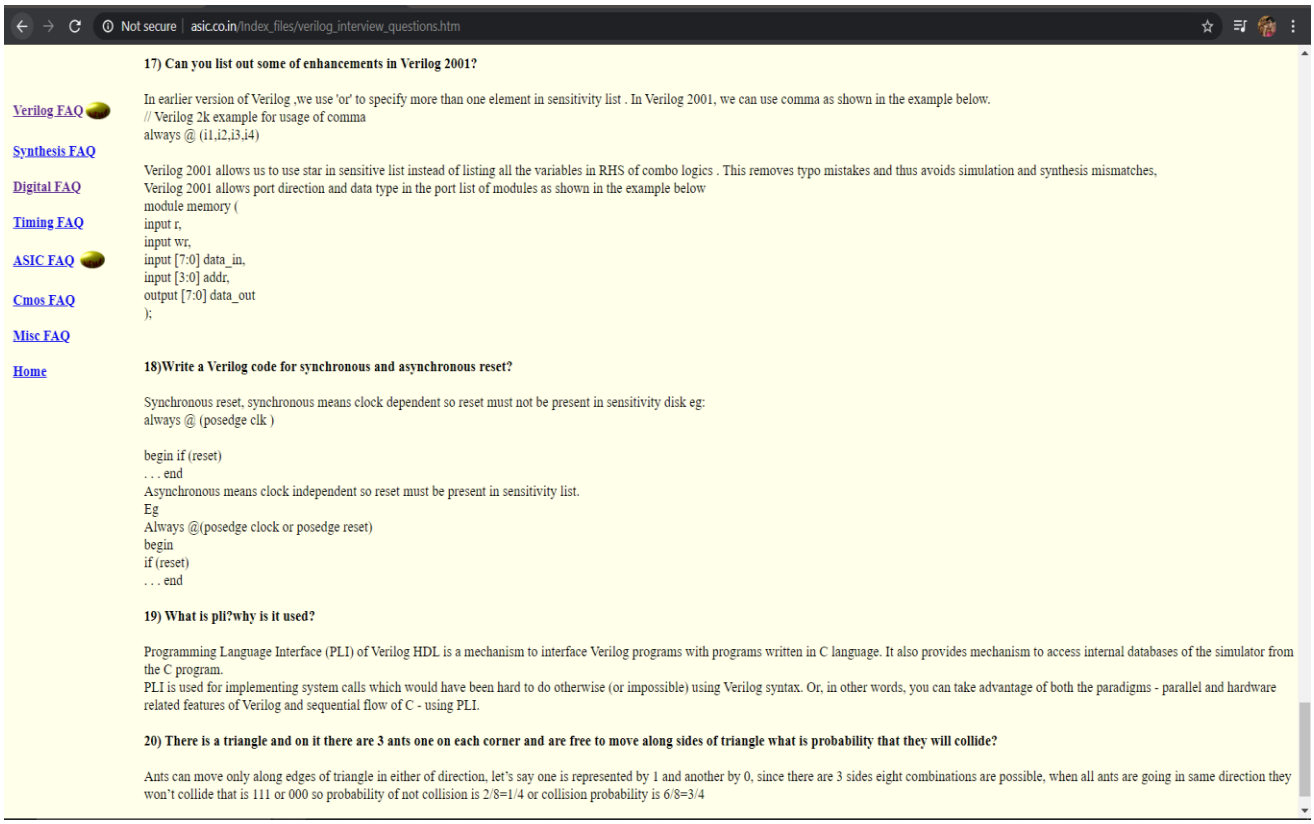
Hardware modelling using Verilog:

- **Learnt about the Verilog hardware description language.**
- **Understood the difference between behavioral and structural design styles.**
- **Learnt to write test benches and analyze simulation results.**
- **Learnt to model combinational and sequential circuits.**
- **Distinguish between good and bad coding practices.**
- **Case studies with some complex designs.**



FPGA and ASIC Interview questions:

- **Gone through the most frequently asked questions from Verilog in Interviews.**



Not secure | asic.co.in/Index_files/verilog_interview_questions.htm

Verilog FAQ
Synthesis FAQ
Digital FAQ
Timing FAQ
ASIC FAQ
Cmos FAQ
Misc FAQ
Home

17) Can you list out some of enhancements in Verilog 2001?

In earlier version of Verilog ,we use 'or' to specify more than one element in sensitivity list . In Verilog 2001, we can use comma as shown in the example below.
// Verilog 2k example for usage of comma
always @ (i1,i2,i3,i4)

Verilog 2001 allows us to use star in sensitive list instead of listing all the variables in RHS of combo logics . This removes typo mistakes and thus avoids simulation and synthesis mismatches,
Verilog 2001 allows port direction and data type in the port list of modules as shown in the example below
module memory (
input r,
input wr,
input [7:0] data_in,
input [3:0] addr,
output [7:0] data_out
);

18)Write a Verilog code for synchronous and asynchronous reset?

Synchronous reset, synchronous means clock dependent so reset must not be present in sensitivity disk eg:
always @ (posedge clk)

begin if (reset)
. . . end
Asynchronous means clock independent so reset must be present in sensitivity list.
Eg
Always @(posedge clock or posedge reset)
begin
if (reset)
. . . end

19) What is pli?why is it used?

Programming Language Interface (PLI) of Verilog HDL is a mechanism to interface Verilog programs with programs written in C language. It also provides mechanism to access internal databases of the simulator from the C program.
PLI is used for implementing system calls which would have been hard to do otherwise (or impossible) using Verilog syntax. Or, in other words, you can take advantage of both the paradigms - parallel and hardware related features of Verilog and sequential flow of C - using PLI.

20) There is a triangle and on it there are 3 ants one on each corner and are free to move along sides of triangle what is probability that they will collide?

Ants can move only along edges of triangle in either of direction, let's say one is represented by 1 and another by 0, since there are 3 sides eight combinations are possible, when all ants are going in same direction they won't collide that is 111 or 000 so probability of not collision is 2/8=1/4 or collision probability is 6/8=3/4

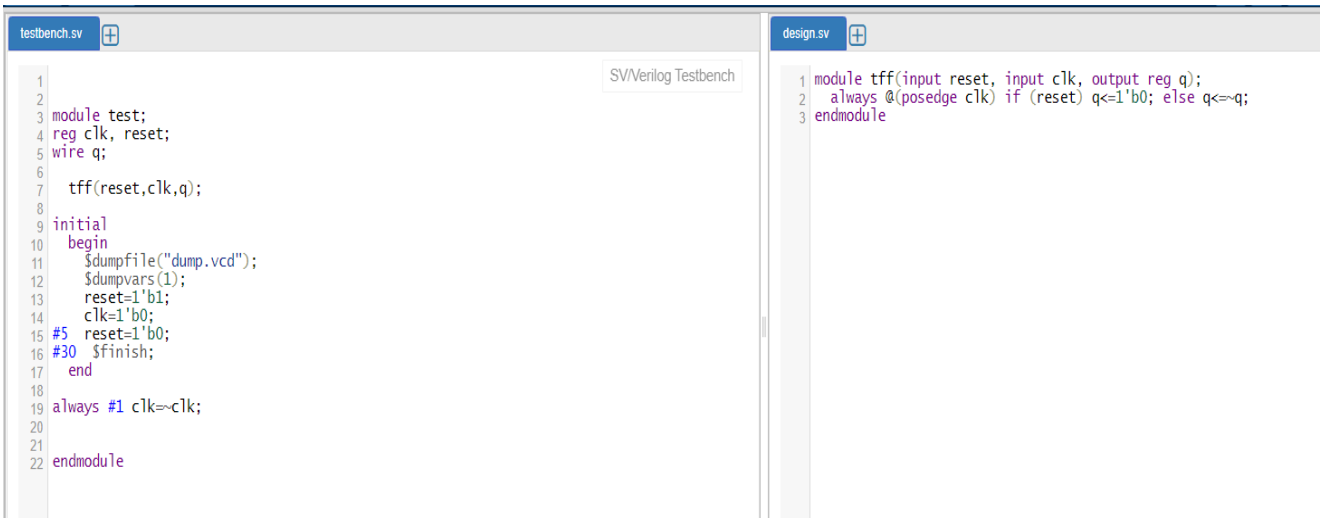
TASK: Implement a simple T Flipflop and test the module using a compiler.

Design:

```
module tff (   input clk,
               input rstn,
               input t,
              output reg q);

  always @ (posedge clk) begin
    if (!rstn)
      q <= 0;
    else
      if (t)
          q <= ~q;
      else
          q <= q;
  end
endmodule
```

Testbench:

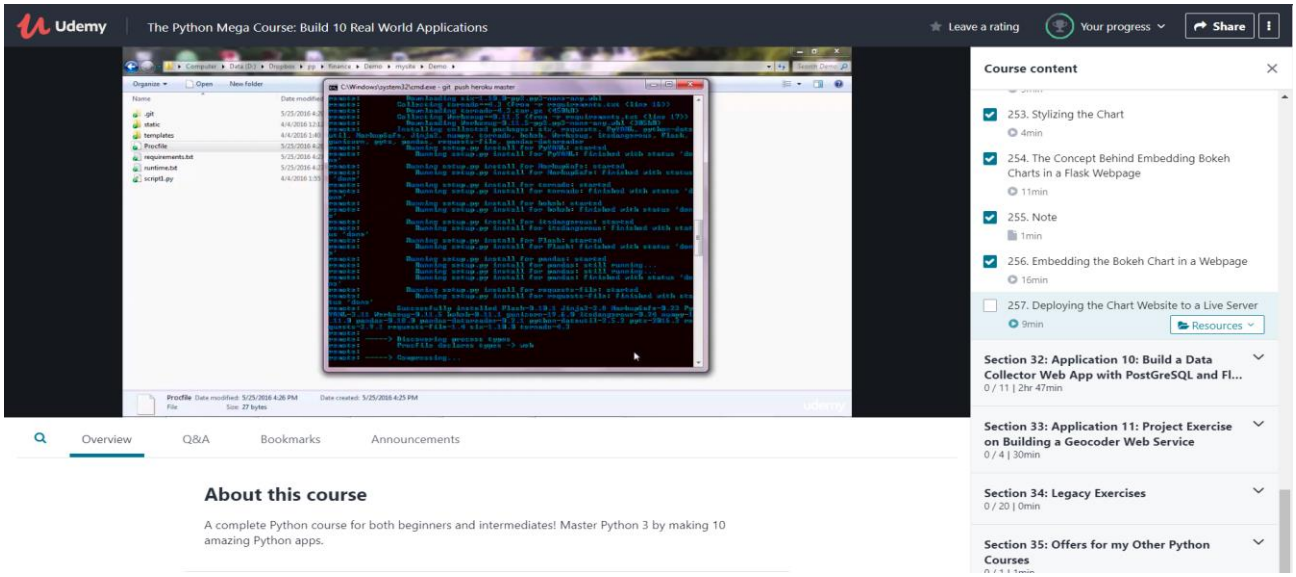
```
module tb;
  reg clk;
  reg rstn;
  reg t;

  tff u0 (  .clk(clk),
            .rstn(rstn),
            .t(t),
          .q(q));

  always #5 clk = ~clk;

  initial begin
    {rstn, clk, t} <= 0;

    $monitor ("T=%0t rstn=%0b t=%0d q=%0d", $time, rstn, t, q);
    repeat(2) @(posedge clk);
    rstn <= 1;

    for (integer i = 0; i < 20; i = i+1) begin
      reg [4:0] dly = $random;
      #(dly) t <= $random;
    end
  #20 $finish;
  end
endmodule
```



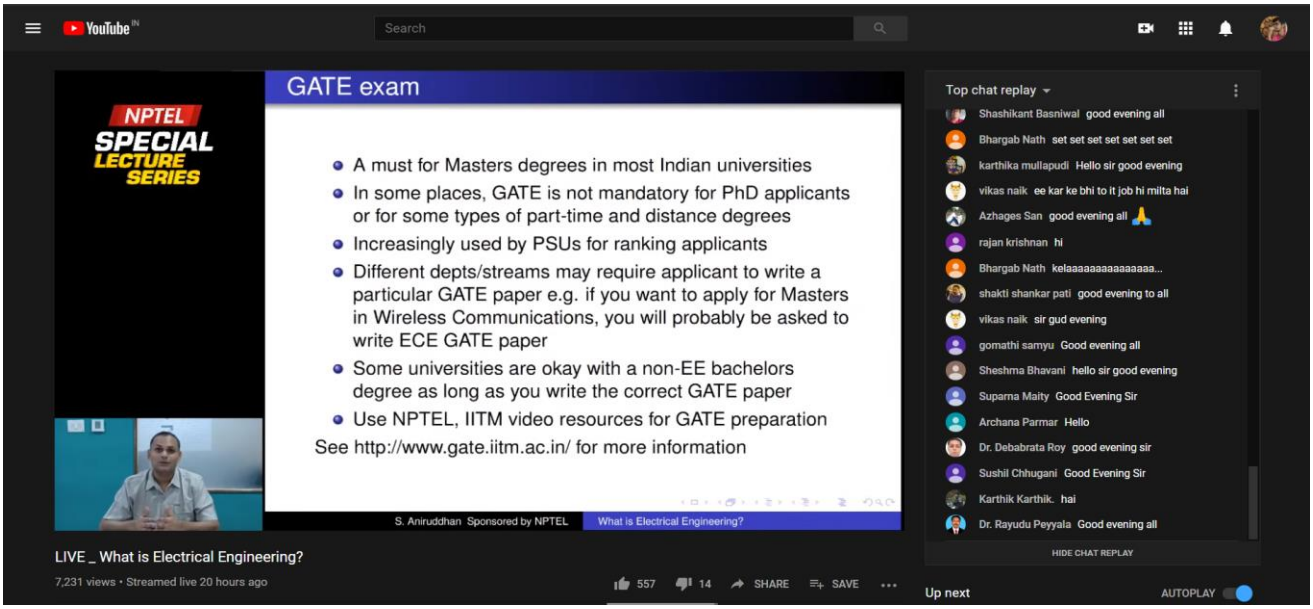
Date:	04 June 2020	Name:	Gagan M K
Course:	The Python Mega Course	USN:	4AL17EC032
Topic:	Application 9: Build a Web-based Financial Graph	Semester & Section:	6th sem & 'A' sec

AFTERNOON SESSION DETAILS

Image of session:



Attended Bonus live session on "What is Electrical Engineering?" by Prof. S.Aniruddhan



Report – Report can be typed or hand written for up to two pages.

Web-based Financial Graph:

- **How to Download Datasets with Python.**
- **Learnt analyzing Stock Market Data.**
- **Plotting Stock Market Data Candlestick Charts.**
- **Updating Candlestick Charts with Bokeh Quadrants**
- **Learnt to plot Candlestick Charts with Bokeh Rectangles**
- **Creating Candlestick Segments**
- **Stylizing the obtained Chart**
- **Learnt the Concept Behind Embedding Bokeh**
- **Sharing the Charts in a Flask Webpage**
- **Learnt how to Embed the Bokeh Chart in a Webpage.**
- **Learnt to Deploy the Chart Website to a Live Server.**

- **Below shown are the some pictures of graph produced, which are the stocks of Google from 01 Jan 2020 to 30 May 2020 and plotted using Candlestick format.**