DAILY ASSESSMENT FORMAT

Date:	**5th June 2020**	**Name:**	**Sushmitha R Naik**
Course:	**Digital design using HDL**	**USN:**	**4AL17EC090**
Topic:	**Verilog Tutorials and practice programs, Building/ Demo projects using FPGA**	**Semester & Section:**	**6th sem 'B' sec**
GitHub Repository:	**Sushmitha_naik**		

FORENOON SESSION DETAILS

Image of session

VERILOG TUTORIALS

- **Verilog is a Hardware Description Language; a textual format for describing electronic circuits and systems. Applied to elect for verification through simulation, for timing analysis, for test analysis (testability analysis and fault gra The Verilog HDL is an IEEE standard - number 1364.**
- **The first version of the IEEE standard for Verilog was published in 1995. A revised version was published in 2001; this is the version used by most Verilog users. The IEEE Verilog standard document is known as the Language Reference Manual, authoritative definition of the Verilog HDL.**
- **A further revision of the Verilog standard was published in 2005, though it has little extra compared to the 2001 standard. System Verilog is a huge set of extensions to Verilog, and was first published as an IEEE standard in 2005. See the ap System Verilog.**
- **IEEE Std 1364 also defines the Programming Language Interface, or PLI. This is a collection of software routines which permit between Verilog and other languages (usual Note that VHDL is not an abbreviation for Verilog HDL - Verilog and VHDL are two different HDLs.**

- **They have more similarities than differences, however. The history of the Verilog HDL goes back to the 1980s, when a company called Gateway Design Automation developed a logic simulator, Verilog-XL, and with it a hardware description language.**
- **Cadence Design Systems acquired Gateway in 1989, and with it the rights to the language and the simulator. In 1990, Cadence put the language (but not the simulator) into the public domain, with the intention that it should become a standard, non-proprietary language.The Verilog HDL is now maintained by a non profit making organisation,**

FPGA

- **FPGA Basics – A Look Under the Hood An introductory look inside Field Programmable Gate Arrays. We'll go over: Strengths & Weaknesses of FPGAs How FPGAs work What's inside an FPGA So you keep hearing about FPGAs being utilized in more and more applications, but aren't sure whether it makes sense to switch to a new technology.**
- **Or maybe you're just getting into the embedded world and want to figure out if an FPGA-based system makes sense for you or not. This paper provides an overview of some of the key elements of FPGAs for engineers interested in utilizing FPGA-based technologies.**
- **It's worth noting that this is a complex topic, and as such, some topics are not covered, some are just introductory, and others will evolve over time.**
- **This paper should still give you a lot of helpful information if you're new to the world of FPGAs.What are the most important things you should know right away? Get out of the software mindset – You're not writing software.**
- **Let me say that again because this is the single most important point if you're thinking about working with FPGAs.You-are-NOTwritingsoftware.You're designing a digital circuit**

T-FLIP FLOP

```
module tff (t, clk, q, qb);
input t, clk;
output q, qb;
reg q, qb;
initial
begin
q=0;
q=1;
end
always@ (posedge (clk))
begin
if(t==0) q=q;
else
q=qb;
qb=~q;
```

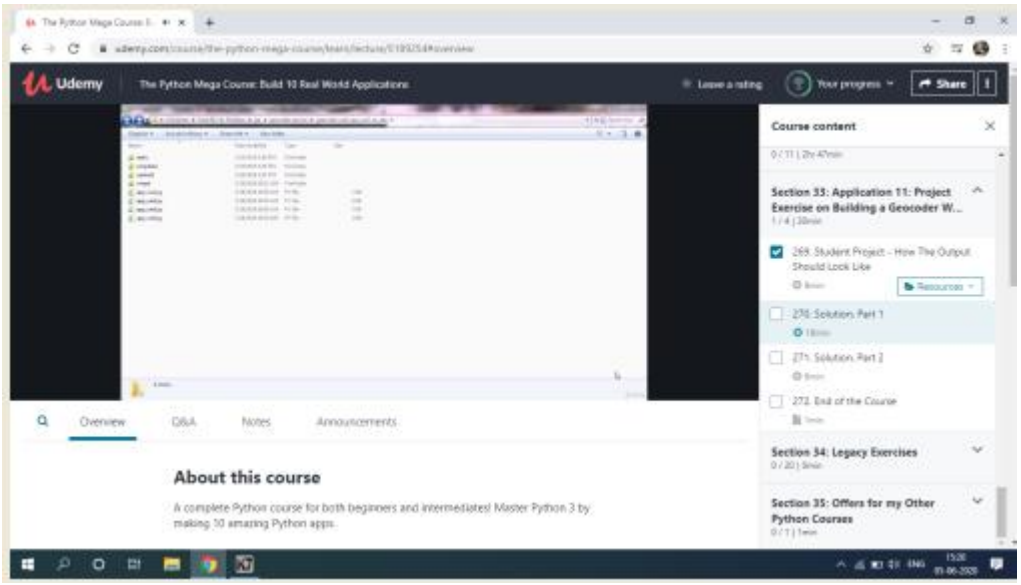
```
end
 endmodule
```

DAILY ASSESSMENT FORMAT

Date:	**5th June 2020**	**Name:**	**Sushmitha R Naik**
Course:	**Python**	**USN:**	**4AL17EC090**
Topic:	**BUILD A DATA COLLECTOR WEB APP WITH POST GRESQL AND FLASK**	**Semester & Section:**	**6th sem 'B' sec**
GitHub Repository:	**Sushmitha_naik**		

AFTERNOON SESSION DETAILS

Image of session



Report:

- **Flask startup and configuration Like most widely used Python libraries, the Flask package is installable from the Python Package Index (PPI).**
- **First create a directory to work in (something like flask_todo is a fine directory name) then install the flask package. You'll also want to install flask-sqlalchemy so your Flask application has a simple way to talk to a SQL database.**
- **A good way to get moving is to turn the codebase into an installable Python distribution. At the project's root, create setup.py and a directory called to-do to hold the source code.**
- **The setup.py should look something like this:**

```
requires = [
          'flask',
         'flask-sqlalchemy',
          'psycopg2',
      ]
```

```
Setup (
        name='flask_todo',
        version='0.0',
        description='A To-Do List built with Flask',
        author='<Your actual name here>',
        author_email='<Your actual e-mail address here>',
        keywords='web flask',
       packages=find_packages (),
       include_package_data=True,
       install_requires=requires
    )
```

- This way, whenever you want to install or deploy your project, you'll have all the necessary packages in the requires list. You'll also have everything you need to set up and install the package in site packages. For more information on how to write an installable Python distribution, check out the docs on setup.py.
- Within the to-do directory containing your source code, create an app.py file and a blank __init__.py file. The __init__.py file allows you to import from to-do as if it were an installed package. The app.py file will be the application's root.
- This is where all the Flask application goodness will go, and you'll create an environment variable that points to that file.
- If you're usingpipenv (like I am), you can locate your virtual environment with pipenv --venv and set up that environment variable in your environment's activate script.