REPORT JUNE 05

Date:	05-06-2020	Name:	Divyashree L V
Course:	Digital Design Using HDL	USN:	4AL17EC030
Topic:	Verilog Tutorials and practice programs,Building/ Demo projects using FPGA , Implement a verilog module to count number of 0’s in a 16 bit number in the compiler.	Semester & Section:	6th sem & A sec
Github Repository:	divyalv		

FORENOON SESSION DETAILS

Image of session

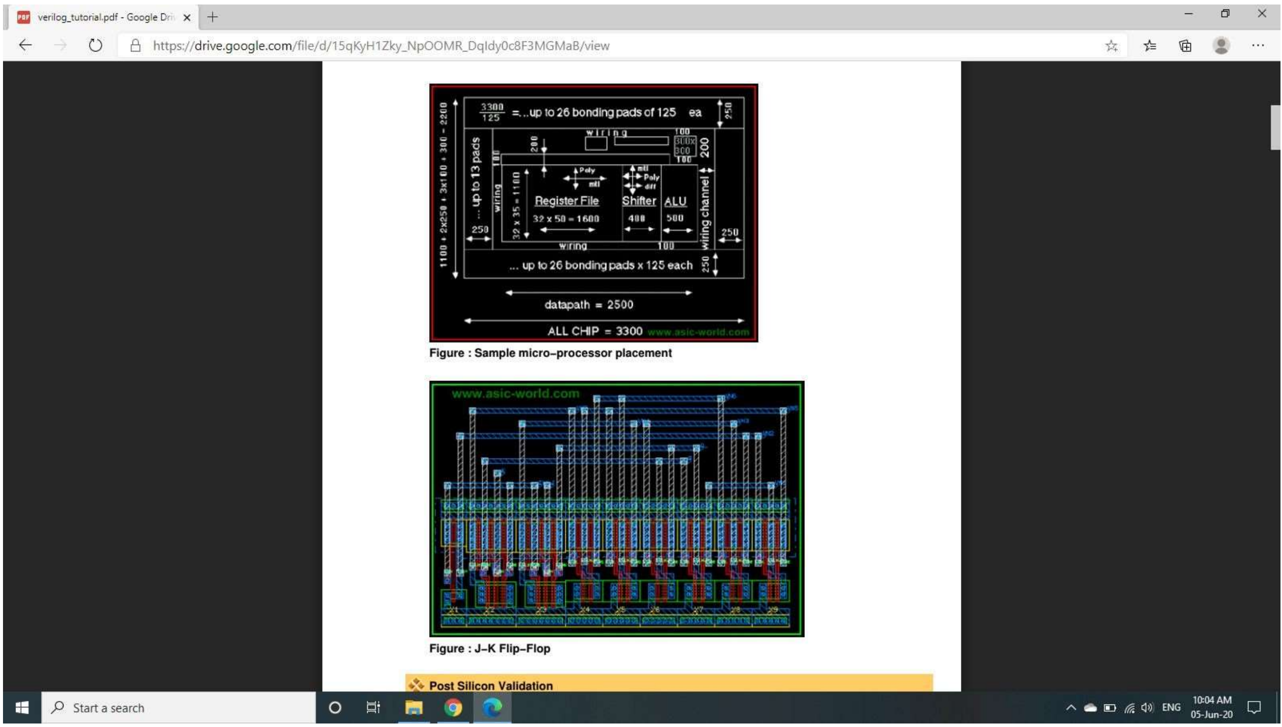

verilog_tutorial.pdf - Google Dri
https://drive.google.com/file/d/15qKyH1Zky_NpOOMR_Dqldy0c8F3MGMaB/view
3300/125 = ...up to 26 bonding pads of 125 ea
wiring
Register File
32 x 50 = 1600
Shifter
ALU
wiring channel
... up to 26 bonding pads x 125 each
datapath = 2500
ALL CHIP = 3300
www.asic-world.com
Start a search
ENG
10:04 AM
05-Jun-20

Figure : Sample micro-processor placement

Figure : J-K Flip-Flop

Post Silicon Validation

Report –

What is HDL ?

A hardware description Language Is a language used to describe a digital system, for example, a network switch, a microprocessor or a memory or a simple flip=flop. This just means that, by using a HDL one can describe any hardware (digital) at any level.

One can describe a simple Flip flop as that in above figure as well as one can describe a complicated designs having 1 million gates. Verilog is one of the HDL languages available in the industry for designing the Hardware. Verilog allows us to design a Digital design at Behavior Level,

Register Transfer Level (RTL), Gate level and at switch level. Verilog allows hardware designers to express their designs with behavioral constructs, deterring the details of implementation to a later stage of design in the final design.

Design Styles:

- **Top Up Design**
- **Bottom Up Design**

Abstract Level of Verilog

- **Behavioral Level**
  This level describes a system by concurrent algorithms (Behavioral). Each algorithm itself is sequential, that means it consists of a set of instructions that are executed one after the other.Functions, Tasks and Always blocks are the main elements. There is no regard to the structural realization of the design.
- **Register Transfer Level**
  Designs using the Register=Transfer Level specify the characteristics of a circuit by operations and the transfer of data between the registers. An explicit clock is used. RTL design contains exact timing possibilities, operations are scheduled to occur at certain times. Modern definition of a RTL code is "Any code that is synthesizable is called RTL code".
- **Gate Level**
  Within the logic level the characteristics of a system are described by logical links and their timing properties. All signals are discrete signals. They can only have definite logical values (`0', `1', `X', `Z`). The usable operations are predefined logic primitives (AND, OR, NOT etc gates). Using gate level modeling might not be a good idea for any level of logic design. Gate level code is generated by tools like synthesis tools and this netlist is used for gate level simulation and for backend.

Implement a verilog module to count number of 0’s in a 16 bit number in compiler.

```
module num_zeros_for(
    input [15:0] A,
    output reg [4:0] ones
    );

integer i;

always@(A)
begin
    ones = 0;
    for(i=0;i<16;i=i+1)
        if(A[i] == 0'b1)
            ones = ones + 1;
end

endmodule
```

output

Input = "1010_0010_1011_0010" => Output = "01001" (9 in decimal)

Input = "0011_0110_1000_1011" => Output = "01000" (8 in decimal)

Date: 05-06- 2020

Course: divyalv

Topic: Geocoder

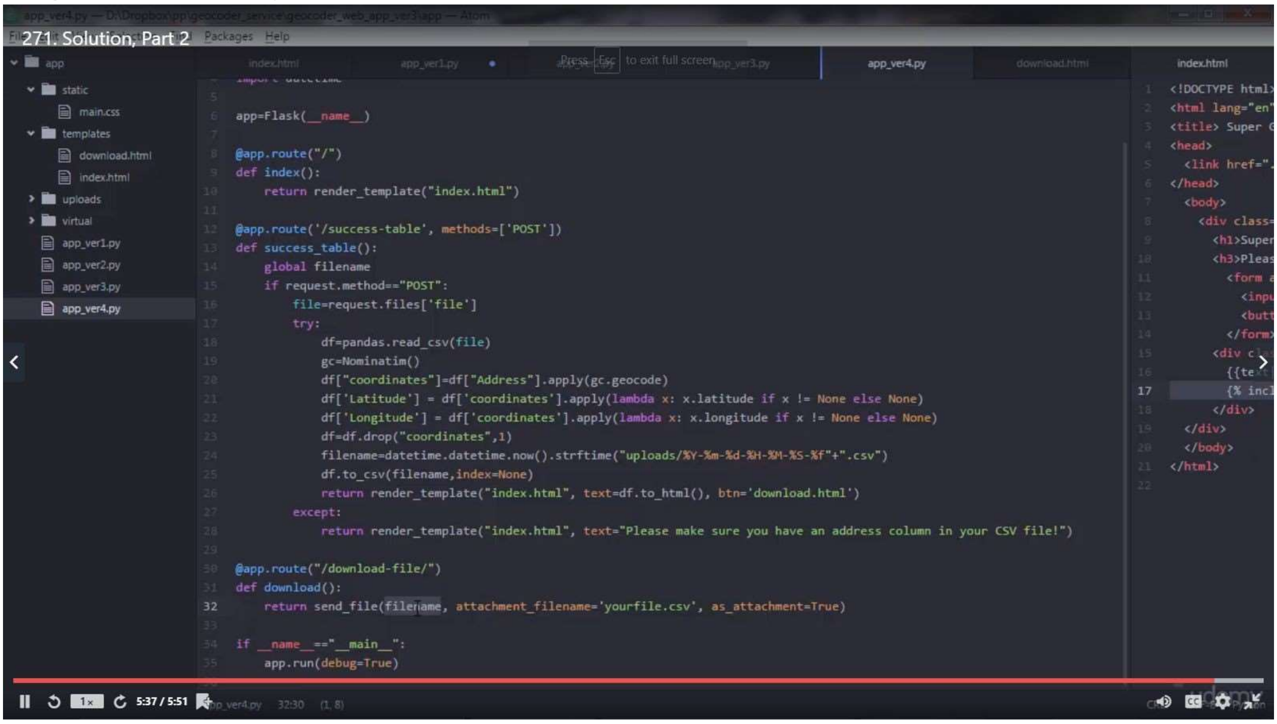
Name: DivyaShree L V

USN: 4AL17EC030

Semester & Section: 6th sem & A sec

AFTERNOON SESSION DETAILS

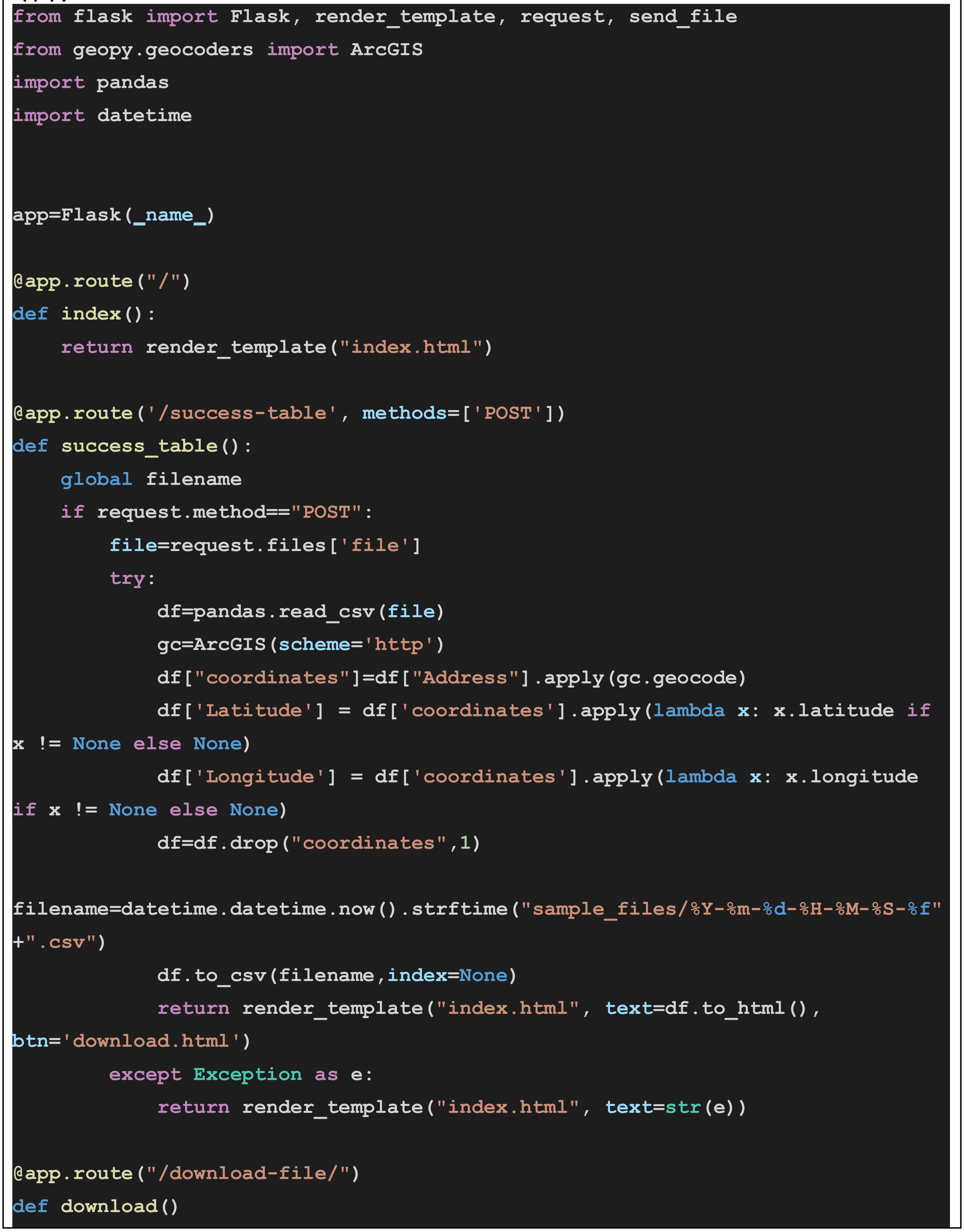
Image of session:Output



```python
app=Flask(__name__)

@app.route("/")
def index():
    return render_template("index.html")

@app.route('/success-table', methods=['POST'])
def success_table():
    global filename
    if request.method=="POST":
        file=request.files['file']
        try:
            df=pandas.read_csv(file)
            gc=Nominatim()
            df["coordinates"]=df["Address"].apply(gc.geocode)
            df['Latitude'] = df['coordinates'].apply(lambda x: x.latitude if x != None else None)
            df['Longitude'] = df['coordinates'].apply(lambda x: x.longitude if x != None else None)
            df=df.drop("coordinates",1)
            filename=datetime.datetime.now().strftime("uploads/%Y-%m-%d-%H-%M-%S-%f"+".csv")
            df.to_csv(filename,index=None)
            return render_template("index.html", text=df.to_html(), btn='download.html')
        except:
            return render_template("index.html", text="Please make sure you have an address column in your CSV file!")

@app.route("/download-file/")
def download():
    return send_file(filename, attachment_filename='yourfile.csv', as_attachment=True)

if __name__=="__main__":
    app.run(debug=True)
```

```html
<!DOCTYPE html>
<html lang="en"
<title> Super
<head>
  <link href="
</head>
  <body>
    <div class=
      <h1>Super
      <h3>Pleas
        <form
          <inpu
          <butt
        </form>
      <div c
        {{te
        {% incl
      </div>
    </div>
  </body>
</html>
```

app.py



```
from flask import Flask, render_template, request, send_file
from geopy.geocoders import ArcGIS
import pandas
import datetime


app=Flask(_name_)

@app.route("/")
def index():
    return render_template("index.html")

@app.route('/success-table', methods=['POST'])
def success_table():
    global filename
    if request.method=="POST":
        file=request.files['file']
        try:
            df=pandas.read_csv(file)
            gc=ArcGIS(scheme='http')
            df["coordinates"]=df["Address"].apply(gc.geocode)
            df['Latitude'] = df['coordinates'].apply(lambda x: x.latitude if
x != None else None)
            df['Longitude'] = df['coordinates'].apply(lambda x: x.longitude
if x != None else None)
            df=df.drop("coordinates",1)

filename=datetime.datetime.now().strftime("sample_files/%Y-%m-%d-%H-%M-%S-%f"
+".csv")
            df.to_csv(filename,index=None)
            return render_template("index.html", text=df.to_html(),
btn='download.html')
        except Exception as e:
            return render_template("index.html", text=str(e))

@app.route("/download-file/")
def download()
```