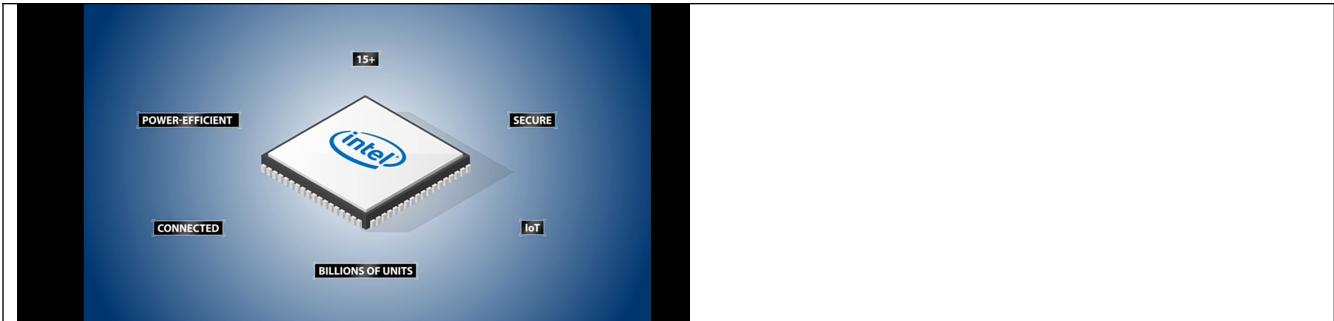


## DAY 12 REPORT

Date:	01/06/20	Name:	ANKITHA C C
Course:	DIGITAL DESIGNING USING HDL	USN:	4AL16EC004
Topic:	ABOUT FPGA AND ASIC	Semester & Section:	8TH & A
Github Repository:	ankitha-course		

FORENOON SESSION DETAILS	
Image of session	A 3D isometric diagram illustrating a networked industrial or smart city system. It features several green rectangular platforms representing different locations. On these platforms are various buildings and structures, such as a wind turbine, a factory building, a residential area, and a bridge. A central server-like unit is connected by lines to all the platforms, indicating a distributed network architecture.
Report - Report can be typed or hand written for up to two pages.	<p>1. Industry application of FPGA</p> <ul style="list-style-type: none"><li>* INTEL</li><li>* IOT</li><li>* CONNECTED</li><li>* Secure</li></ul>



## 2. FPGA BUSINESS FUNDAMENTAL

**FPGA Business Fundamentals**  
Intel PSG, CEG

**Course Summary**

- ASIC vs. ASSP vs. FPGA
- FPGA's broad customer base
- Benefits of using an FPGA: reprogrammability, product longevity, reduced TTM, market-size optimized
- Importance of Quartus
- Future with Intel

**② FPGA Business Fundamentals**

Different hardware

- ① ASIC - Custom logic
- ② ASSP - Off-the-shelf logic
- ③ FPGA - Logic board

**ASIC/ASSP Advantages**

PROS	CONS
low cost	high cost (NRE)
low power consumption	not flexible
small size	long time to market
high performance	complex design flow

**Why FPGA**

- \* Reprogrammable & flexible
- \* Product longevity
- \* Reduced Time-to-market
- \* Market size optimized

**Broad Customer Base**

The graph illustrates the distribution of customer volumes. The Y-axis represents the percentage of customers, and the X-axis represents the number of customers. The 'ASIC' curve shows a few large, high-volume customers ('head') and very few smaller ones ('tail'). The 'FPGA' curve shows a much larger number of customers, with a significant portion being small to medium-sized ('long tail'), indicating a broader customer base.

**Summary**

- ASIC vs. ASSPs vs. FPGA
- FPGA's broad customer base
- Benefits of using an FPGA: reduced TTM
- Importance of Quartus
- Future with Intel

## 3. FPGA vs ASIC DESIGN FLOW

**FPGA versus ASIC Design Flow**

**(3) FPGA vs ASIC Design Flow**

Key DIO: ASIC & FPGA

- design methodology
- verification techniques
- test generation logic
- tools

**Functional spec**

**ASIC Imp?**

- ① Create HDL → optimized for ASIC technology & reuse
- ② Synthesise → primary driven by script
- ③ Place & route

**FPGA Imp?**

- ① Create HDL → xilinx
- ② Synthesise → memory, XST
- ③ Place & route → ISE

## 4. 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

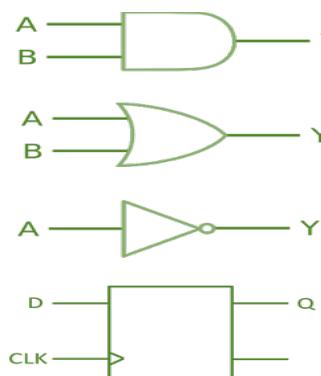
So you keep hearing about FPGAs being utilized in more and more applications, but aren't sure whether it makes sense to switch to a new technology. Or maybe you're just getting into the embedded world and want to figure out if an FPGA-based system makes sense for you or not.

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.



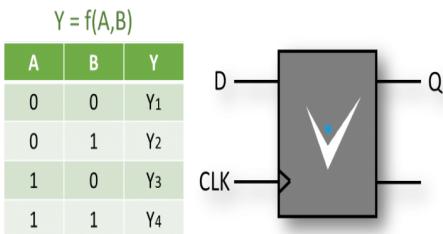
How Does an FPGA work?

FPGA-basics-gates-and-flip-flops



You're designing a digital circuit more than anything else, basically at one layer of abstraction above the logic gate (AND, OR, NOT) level. At the most basic level, you need to think about how you're specifying the layout and equations

at the level of LUTs (Look-Up Tables) and FFs (Flip-Flops).



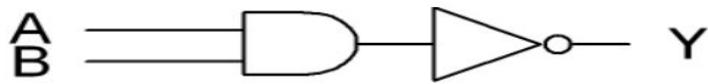
Otherwise your circuit can get very large and slow very quickly. You've got a very detailed level of control at your fingertips, which is very powerful, but can be overwhelming, so start slow. You'll be determining the # of bits, and exact math / structure of each function.

An FPGA is a synchronous device, meaning that logical operations are performed on a clock cycle-by-cycle basis. Flip-flops are the core element to enabling this structure.

### TASK 1

#### VERILOG CODE TO IMPLEMENT NAND GATE USING ALL MODELLING :

##### 1. USING GATE LEVEL MODELLING



```
module NAND_2_gate_level(output Y, input A, B);
    wire Yd;
    and(Yd, A, B);
    not(Y, Yd);
endmodule
```

##### 2. USING DATA FLOW MODELLING

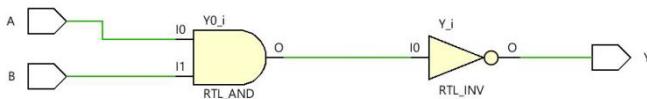
The boolean equation for a NAND gate is  $Y = (A \cdot B)'$  or  $\sim(A \& B)$ .

```
module NAND_2_data_flow (output Y, input A, B);
    assign Y = ~(A & B);
endmodule
```

### 3. BEHAVIORAL MODELLING

```
module NAND_2_behavioral (output reg Y, input A, B);  
always @ (A or B) begin  
if (A == 1'b1 & B == 1'b1) begin  
Y = 1'b0;  
end  
else  
Y = 1'b1;  
end  
endmodule
```

### 4. RTL schematic of the NAND gate

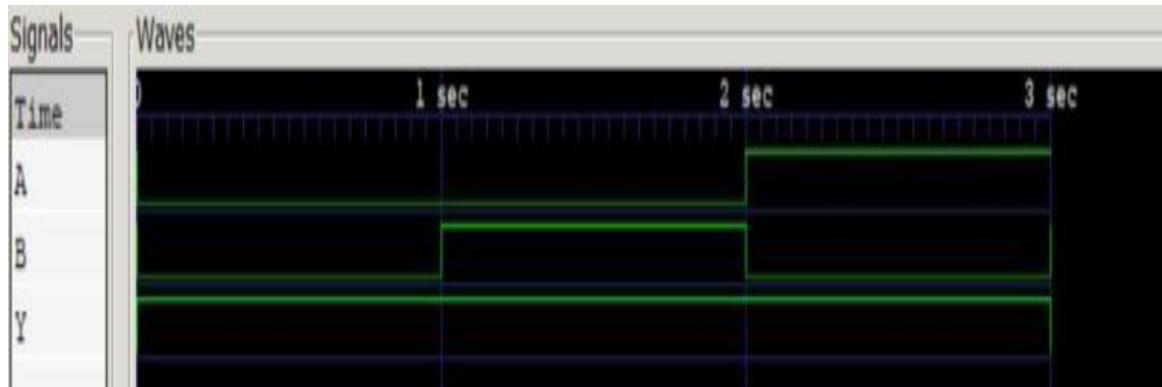


### Testbench of the NAND gate using Verilog

```
'include "NAND_2_behavioral.v"  
module NAND_2_behavioral_tb;  
reg A, B;  
wire Y;  
NAND_2_behavioral instance0 (Y, A, B);  
initial begin  
A = 0; B = 0;  
#1 A = 0; B = 1;  
#1 A = 1; B = 0;  
#1 A = 1; B = 1;  
end  
initial begin  
$monitor ("%t | A = %d| B = %d| Y = %d", $time, A, B, Y);
```

```
$dumpfile("dump.vcd");
$dumpvars();
end
endmodule
```

Simulation Waveform



Date: 01/06/20

Name: Ankitha c c

Course: Python

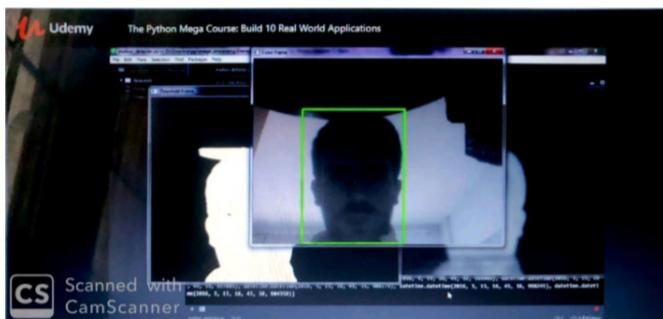
USN: 4al16ec004

Topic:

Semester & 8th A  
Section:

### AFTERNOON SESSION DETAILS

Image of session



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

WebCam Motion Detector in Python

This python program will allow you to detect motion and also store the time interval of the motion.

Requirement:

Python3

OpenCV(libraries)

Pandas(libraries)

Main Logic : Videos can be treated as stack of pictures called frames. Here I am comparing different frames(pictures) to the first frame which should be static(No movements initially). We compare two images by comparing the intensity value of each pixels. In python we can do it easily as you can see in following code:

```
filter_none
edit
play_arrow
brightness_4
# Python program to implement
# Webcam Motion Detector

# importing OpenCV, time and Pandas library
import cv2, time, pandas
# importing datetime class from datetime library
from datetime import datetime

# Assigning our static_back to None
static_back = None

# List of moving objects
motion_list = [None, None]

# Time of movement
time = []

# Initializing DataFrame, one column is start
# time and other column is end time
df = pandas.DataFrame(columns = ["Start", "End"])

# Capturing video
video = cv2.VideoCapture(0)

# Infinite loop to treat stack of image as video
while True:
```

```
# Reading frame(image) from video
check, frame = video.read()

# Initializing motion = 0(no motion)
motion = 0

# Converting color image to gray_scale image
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

# Converting gray scale image to GaussianBlur
# so that change can be find easily
gray = cv2.GaussianBlur(gray, (21, 21), 0)

# In first iteration we assign the value
# of static_back to our first frame
if static_back is None:
    static_back = gray
    continue

# Difference between static background
# and current frame(which is GaussianBlur)
diff_frame = cv2.absdiff(static_back, gray)

# If change in between static background and
# current frame is greater than 30 it will show white color(255)
thresh_frame = cv2.threshold(diff_frame, 30, 255, cv2.THRESH_BINARY)[1]
thresh_frame = cv2.dilate(thresh_frame, None, iterations = 2)

# Finding contour of moving object
cnts,_ = cv2.findContours(thresh_frame.copy(),
```

```
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

```
for contour in cnts:
```

```
    if cv2.contourArea(contour) < 10000:
```

```
        continue
```

```
    motion = 1
```

```
    (x, y, w, h) = cv2.boundingRect(contour)
```

```
    # making green rectangle arround the moving object
```

```
    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 3)
```

```
# Appending status of motion
```

```
motion_list.append(motion)
```

```
motion_list = motion_list[-2:]
```

```
# Appending Start time of motion
```

```
if motion_list[-1] == 1 and motion_list[-2] == 0:
```

```
    time.append(datetime.now())
```

```
# Appending End time of motion
```

```
if motion_list[-1] == 0 and motion_list[-2] == 1:
```

```
    time.append(datetime.now())
```

```
# Displaying image in gray_scale
```

```
cv2.imshow("Gray Frame", gray)
```

```
# Displaying the difference in currentframe to
```

```
# the staticframe(very first_frame)
```

```
cv2.imshow("Difference Frame", diff_frame)
```

```

# Displaying the black and white image in which if
# intensity difference greater than 30 it will appear white
cv2.imshow("Threshold Frame", thresh_frame)

# Displaying color frame with contour of motion of object
cv2.imshow("Color Frame", frame)

key = cv2.waitKey(1)
# if q entered whole process will stop
if key == ord('q'):
    # if something is moving then it append the end time of movement
    if motion == 1:
        time.append(datetime.now())
        break

# Appending time of motion in DataFrame
for i in range(0, len(time), 2):
    df = df.append({"Start":time[i], "End":time[i + 1]}, ignore_index = True)

# Creating a CSV file in which time of movements will be saved
df.to_csv("Time_of_movements.csv")

video.release()

# Destroying all the windows
cv2.destroyAllWindows()

Analysis of all windows
After running the code there 4 new window will appear on screen. Let's analyse it one by one:
Gray Frame : In Gray frame the image is a bit blur and in grayscale we did so because, In gray pictures there is only one intensity value

```

whereas in RGB(Red, Green and Blue) image there are three intensity values. So it would be easy to calculate the intensity difference in grayscale.



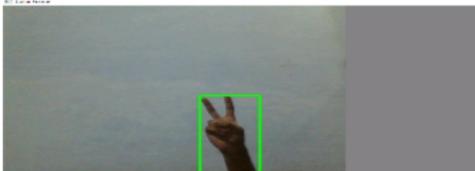
2. **Difference Frame** : Difference frame shows the difference of intensities of first frame to the current frame.



3. **Threshold Frame**: If the intensity difference for a particular pixel is more than 30(in my case) then that pixel will be white and if the difference is less than 30 that pixel will be black



4. **Color Frame** : In this frame you can see the color images in color frame along with green contour around the moving objects



### **Time Record of movements**

The Time\_of\_movements file will be stored in the folder where your code file is stored. This file will be in csv extension. In this file the start time of motion and the end time of motion

will be recorded. As you can see in picture:





