**29 May 2020**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Date:** | | **29 May 2020** | | | **Name:** | **Srinidhi J C** |
| **Course:** | | **Logic Design** | | | **USN:** | **4al16ec078** |
| **Topic:** | | |  | | --- | | **Analysis of clocked sequential circuits** | | **Digital clock design** | | | | **Semester & Section:** | **8th -Sem, B-Sec** |
| **Github Repository:** | | **SrinidhiJC078** | | |  |  |
| **FORENOON SESSION DETAILS** | | | | | | |
| **Image of session**  **A screen shot of a computer  Description automatically generated** | | | | | | |
| Report – Report can be typed or hand written for up to two pages.  **Sequential circuit** combinational logic circuit that consists of inputs variable (X), logic gates (Computational circuit), and output variable (Z).    Combinational circuit produces an output based on input variable only, but **Sequential circuit** produces an output based on **current input and previous input variables**. That means sequential circuits include memory elements which are capable of storing binary information. That binary information defines the state of the sequential circuit at that time. A latch capable of storing one bit of information.    As shown in figure there are two types of input to the combinational logic :   1. External inputs which not controlled by the circuit. 2. Internal inputs which are a function of a previous output states.   Secondary inputs are state variables produced by the storage elements, where as secondary outputs are excitations for the storage elements.  **Types of Sequential Circuits –** There are two types of sequential circuit :   1. **Asynchronous sequential circuit –** These circuit **do not use a clock signal** but uses the pulses of the inputs. These circuits are **faster** than synchronous sequential circuits because there is clock pulse and change their state immediately when there is a change in the input signal. We use asynchronous sequential circuits when speed of operation is important and **independent** of internal clock pulse.     But these circuits are more **difficult** to design and their output is **uncertain**.   1. **Synchronous sequential circuit –** These circuit **uses clock signal** and level inputs (or pulsed) (with restrictions on pulse width and circuit propagation). The output pulse is the same duration as the clock pulse for the clocked sequential circuits. Since they wait for the next clock pulse to arrive to perform the next operation, so these circuits are bit **slower** compared to asynchronous. Level output changes state at the start of an input pulse and remains in that until the next input or clock pulse.     The main parts of the circuit are as follows:  1- **Timer 555**: Responsible for generating the clock pulses for the counters, the frequency of the output shoul be 1 hz which means 1 second for each pulse.  2- **Counters**: Responsible for generating the time in BCD (Binary Coded decimal).  3- **Decoders** : Takes the BCD of the counter as input and produces 7 segment output .  4- 7 **segments** : Displays the time, of course!  \* Seconds have 2 displays , 2 decoders and 2 counters. The same for minutes and hours.  One thing you should know before explaining how the circuit works is that 7490 decade counters  respond only to the negative going edge of 555 pulses, which means it will change its state only when the clock goes from 1 to 0.  **The circuit works as follows :**  555 timer produces 1 seconde pulses to the clock input of the first counter which is responsible the first column of seconds, so its output will change every second.  The counter produces numbers from 0 to 9 in BCD form and automatically resets to 0 after that.  so the output of the first counter will count from 0 to 9 every second and that's exactly what we want from it, so we are done here. let's move to the next one.  **First**, we want the 2nd counter to start counting when the 1st one moves for 9 to 0 (that makes 10 seconds!)  **How can this be done?**  let's check the output of the fist counter in BCD :  MSB---LSB  0:  0000  1:  0001  2:  0010  3:  0011  4:  0100  5:  0101  6:  0110  7:  0111  8:  1000  9:  1001  0:   0000  Remember that 7490 decade counters  respond only to the pulses that go from 1 to 0 and notice that this case only happens in the BCD code above when the output changes from 9 to 0 (the Most significant bit changes from 1 to 0). So, we'll just connect the clock input of the 2nd counter to the most significant bit of the output of the first counter.  **Second**, Since we have 60 seconds in the minute we want the 2nd counter to count only to 5, that makes 59 seconds maximum, when it take another pulse it doesn't count to 60, instead it resets itself to 0 and send a pulse to the first counter in minutes to tell it to count 1 minute | | | | | | |
| **Date:** | **29 May 2020** | | **Name:** | **Srinidhi J C** | | |
| **Course:** | **Python** | | **USN:** | **4al16ec078** | | |
| **Topic:** | **Python for Image and Video processing with OpenCV** | | **Semester & Section:** | **8th-Sem, B-Sec** | | |
| **AFTERNOON SESSION DETAILS** | | | | | | |
| **Image of session**  **A screenshot of a computer screen  Description automatically generatedA screenshot of a cell phone screen with text  Description automatically generatedA screenshot of a cell phone screen with text  Description automatically generated**  Report – Report can be typed or handwritten for up to two pages.  [**OpenCV**](https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html)(Open source computer vision) is a [library of programming functions](https://en.wikipedia.org/wiki/Library_(computing)) mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision). Several IoT and Machine learning techniques can be done by it. It’s one of the most powerful computer visions.  **Program for Face Detection.**  import cv2    # create a new cam object  cap = cv2.VideoCapture(0)  # initialize the face recognizer (default face haar cascade)  face\_cascade = cv2.CascadeClassifier("cascades/haarcascade\_fontalface\_default.xml")  while True:  # read the image from the cam  \_, image = cap.read()  # converting to grayscale  image\_gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)  # detect all the faces in the image  faces = face\_cascade.detectMultiScale(image\_gray, 1.3, 5)  # for every face, draw a blue rectangle  for x, y, width, height in faces:  cv2.rectangle(image, (x, y), (x + width, y + height), color=(255, 0, 0), thickness=2)  cv2.imshow("image", image)  if cv2.waitKey(1) == ord("q"):  break  cap.release()  cv2.destroyAllWindows() Face Recognition: Every Machine Learning algorithm takes a dataset as input and learns from this data. The algorithm goes through the data and identifies patterns in the data. For instance, suppose we wish to identify whose face is present in a given image, there are multiple things we can look at as a pattern:   * Height/width of the face. * Height and width may not be reliable since the image could be rescaled to a smaller face. However, even after rescaling, what remains unchanged are the ratios – the ratio of height of the face to the width of the face won’t change. * Color of the face. * Width of other parts of the face like lips, nose, etc.   face recognition library in Python can perform a large number of tasks:   * Find all the faces in a given image * Find and manipulate facial features in an image * Identify faces in images * Real-time face recognition   After detecting faces, the faces can also be recognized and the object/Person name can notified above.  **Program for video capturing:**  import cv2  #Capture video from webcam  vid\_capture = cv2.VideoCapture(0)  vid\_cod = cv2.VideoWriter\_fourcc(\*'XVID')  output = cv2.VideoWriter("videos/cam\_video.mp4", vid\_cod, 20.0, (640,480))  while(True):  # Capture each frame of webcam video  ret,frame = vid\_capture.read()  cv2.imshow("My cam video", frame)  output.write(frame)  # Close and break the loop after pressing "x" key  if cv2.waitKey(1) &0XFF == ord('x'):  break  # close the already opened camera  vid\_capture.release()  # close the already opened file  output.release()  # close the window and de-allocate any associated memory usage  cv2.destroyAllWindows() | | | | | | |

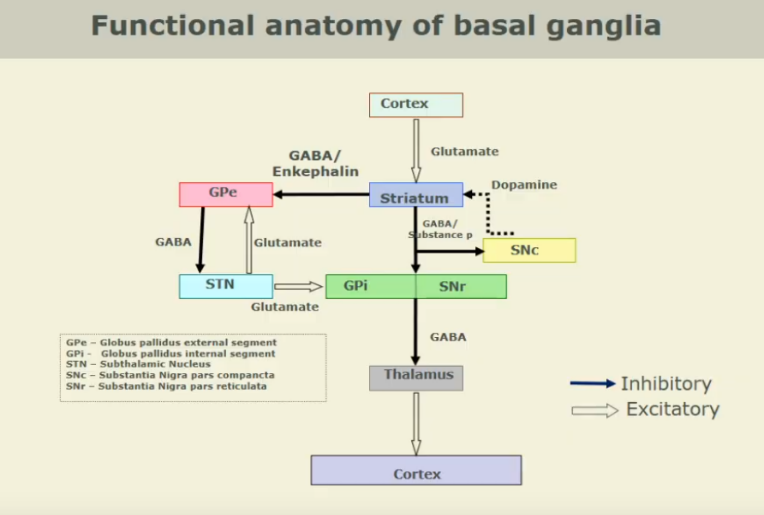
# **Bonus Video:** Live\_Simplifying the Brain by V. Srinivasa Chakravarthy

A screenshot of a cell phone

Description automatically generatedAbout the Topic: In popular media and also mainstream neuroscience literature, there is a tendency to present brain as an extremely complex organ. This talk tries to argue that if you look at the brain in terms of the fundamental principles, it appears reasonably simple. We will then understand that the brain is simple in its underlying principles and complex only in the detaA screenshot of a cell phone

Description automatically generatedils of its organization.

**A circuit board

Description automatically generated**A screenshot of a cell phone

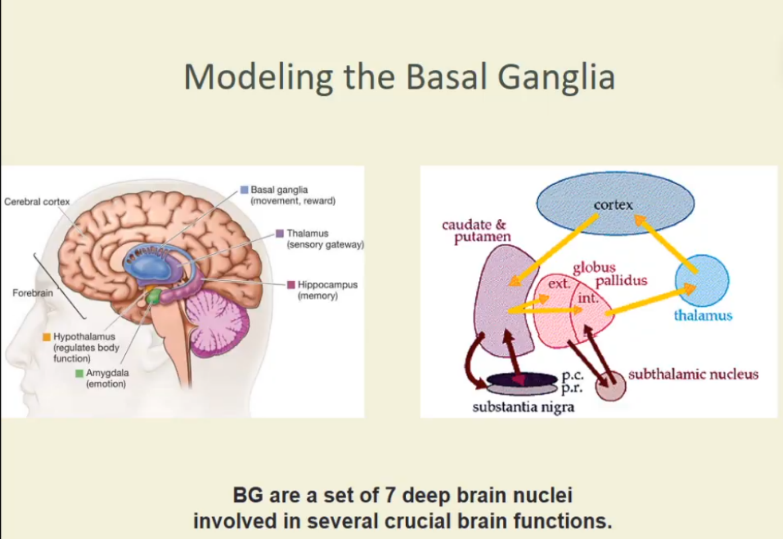
Description automatically generatedA picture containing star, device

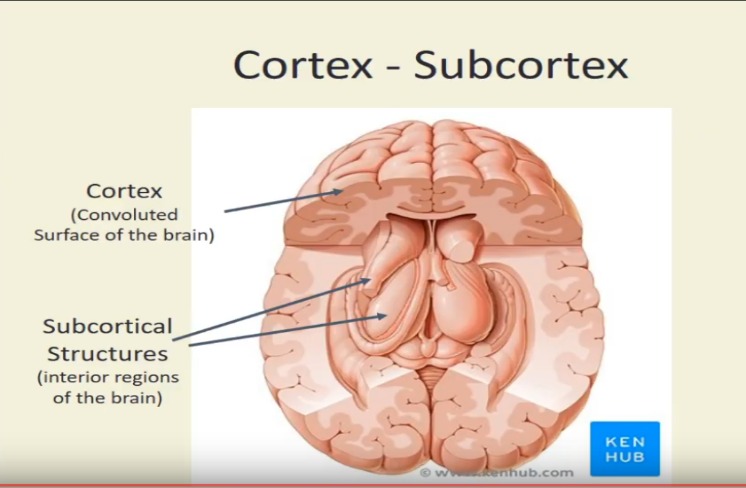
Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a social media post

Description automatically generated

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generatedA screenshot of text

Description automatically generatedA screenshot of a cell phone

Description automatically generated**