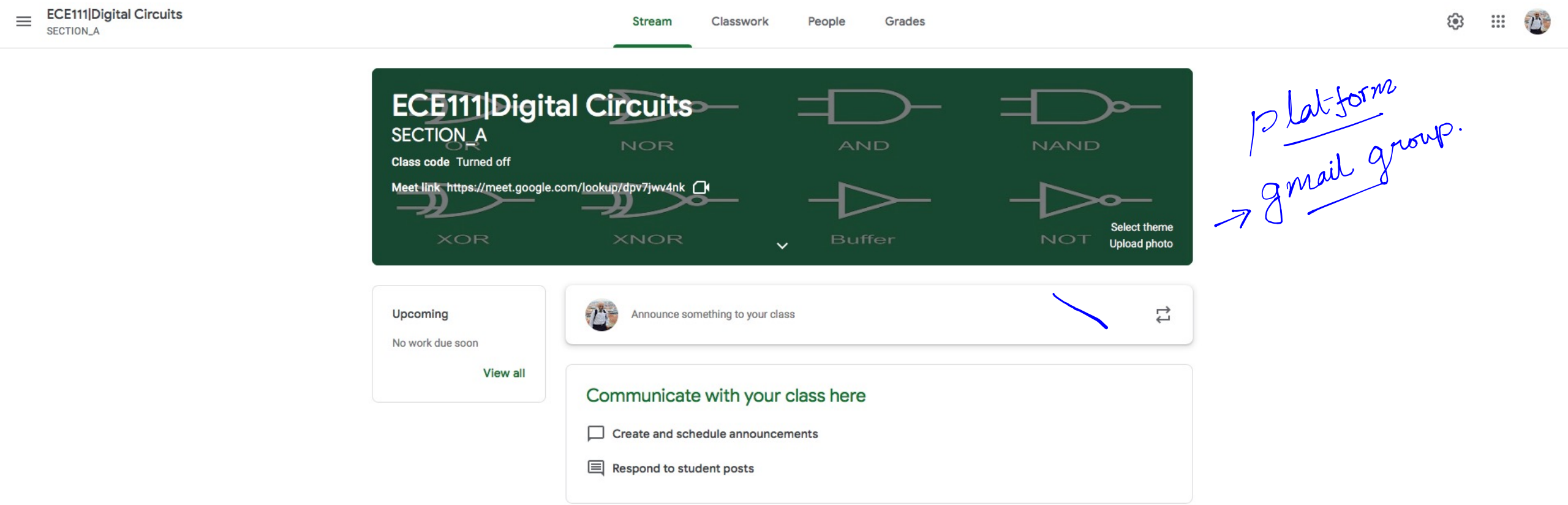
GOOGLE CLASSROOM



platform
→ gmail group.

Responsibility

- Lectures -> Instructor
- Tutorials -> Instructor + Teaching Fellow + Teaching Assistants
- Labs -> Teaching Fellows + Teaching Assistants
- Quizzes and Theory Exams -> Instructor + TF
- Grades -> Students

Teaching Fellow: Khagendra Joshi <khagendra@iiitd.ac.in>

Teaching Assistants:

PhD20111	poojabe@iiitd.ac.in	Pooja Beniwal
PhD21109	shanvis@iiitd.ac.in	Shanvi Srivastava
MT21003	abhipray21003@iiitd.ac.in	Abhipray Agarwal
MT21016	ashwani21016@iiitd.ac.in	Ashwani Dongre
MT21023	devansh21023@iiitd.ac.in	Devansh Shukla
MT21069	risabh21069@iiitd.ac.in	Risabh Sinha
MT21099	suraj21099@iiitd.ac.in	Suraj Singh
MT21152	ekta21152@iiitd.ac.in	Ekta Tiwari
MT21155	rohit21155@iiitd.ac.in	Mate Rohit Prabhakar
MT21157	vaishnav21157@iiitd.ac.in	Nk Vaishnav
MT21158	priyanka21158@iiitd.ac.in	Priyanka Bhagat
MT21159	sarthak21159@iiitd.ac.in	Sarthak Walecha
MT21160	shubham21160@iiitd.ac.in	Shubham Goyal
MT21168	dikshant21168@iiitd.ac.in	Dikshant Yadav

MT21174	saksham21174@iiitd.ac.in	Saksham Girotra
MT21176	swarnima21176@iiitd.ac.in	Swarnima Rawat
MT21182	aadithya21182@iiitd.ac.in	Aadithya Manoharan
MT21185	akash21185@iiitd.ac.in	Akash Saxena
MT21190	deepanshi21190@iiitd.ac.in	Deepanshi
MT21191	diptanu21191@iiitd.ac.in	Diptanu Som
MT21192	justin21192@iiitd.ac.in	Justin Alex
MT21198	naladimmu21198@iiitd.ac.in	Naladimmu Kartheek Ram Reddy
MT21201	pranjal21201@iiitd.ac.in	Pranjal Rajan
MT21203	prince21203@iiitd.ac.in	Prince Kumar Rauniyar
MT21205	ranjith21205@iiitd.ac.in	Ranjith Srinivas A B
MT21206	sachi21206@iiitd.ac.in	Sachi Garg
MT21211	shubham21211@iiitd.ac.in	Shubham Pandey
MT21212	sudhanshu21212@iiitd.ac.in	Sudhanshu Trivedi
MT21213	tarun21213@iiitd.ac.in	Tarun Aggarwal

29 TAs

Evaluation

- Mid-semester Exam: 20%
- Final Exam: 40%
- Assignments: 10%
- Labs: 15%
- Quiz: 15% (Best N-1 out of N)

Evaluation

Plagiarism: Zero tolerance!

https://www.iiitd.ac.in/sites/default/files/docs/education/AcademicDishonesty.pdf

1) **One letter grade reduction for first instance (over entire stay at IIITD) and report to academic department**
2) **F grade for every subsequent instance(s)**

* **In case of habitual offenders in multiple courses or past history, stringent actions will be taken, including loss of a year..**

Evaluation

- Assignments: 10%
- Quiz: 15% (Best N-1 out of N)

- If there is a large disparity in the marks obtained by a student in an assignment and the Quiz following the assignment submission, steps will be taken to check the understanding of the student with respect to the submitted assignment.
- This may be done through a viva or another quiz (announced with a notice of less than a day) for students with large disparity in the marks obtained in an Assignment and corresponding Quiz. If such students continue to perform below par in the exploratory test, the marks in both the Assignment and the Quiz following will be made zero.
- Students with two such zeros in Assignment and Quiz will not be permitted to sit for the end-sem examination.

Open Book Theory Exams and Quizzes

- Only Book and notes written in their own handwriting (no XEROX, printouts)
- Copying solutions from internet will be considered plagiarism

Textbooks

- ***Digital Design with an Introduction to the Verilog HDL* – M. Morris Mano & Michael D. Ciletti, Ed-5, Pearson (Prentice-Hall).**
- Digital Design: Principles and Practices, Fourth Edition, John F. Wakerley, Pearson (Prentice Hall).

Economy Edition.
are also available as pdf in the net.

Attendance Policy

- **Lectures:** Not mandatory/Not Monitored
- **Tutorials:** If attendance before mid-sem is less than 75%, you will get zero marks in one of the quiz and it will be considered as compulsory quiz. If attendance after mid-sem is less than 75%, you will get zero marks in one of the quiz and it will be considered as compulsory quiz.
- **Labs:** All experiments must be completed before each lab-exam. Otherwise, you will not be allowed to appear for lab exams.

Relevance of ECE course for Computer Science:

Computer: Architecture and Organization

Computer Architecture

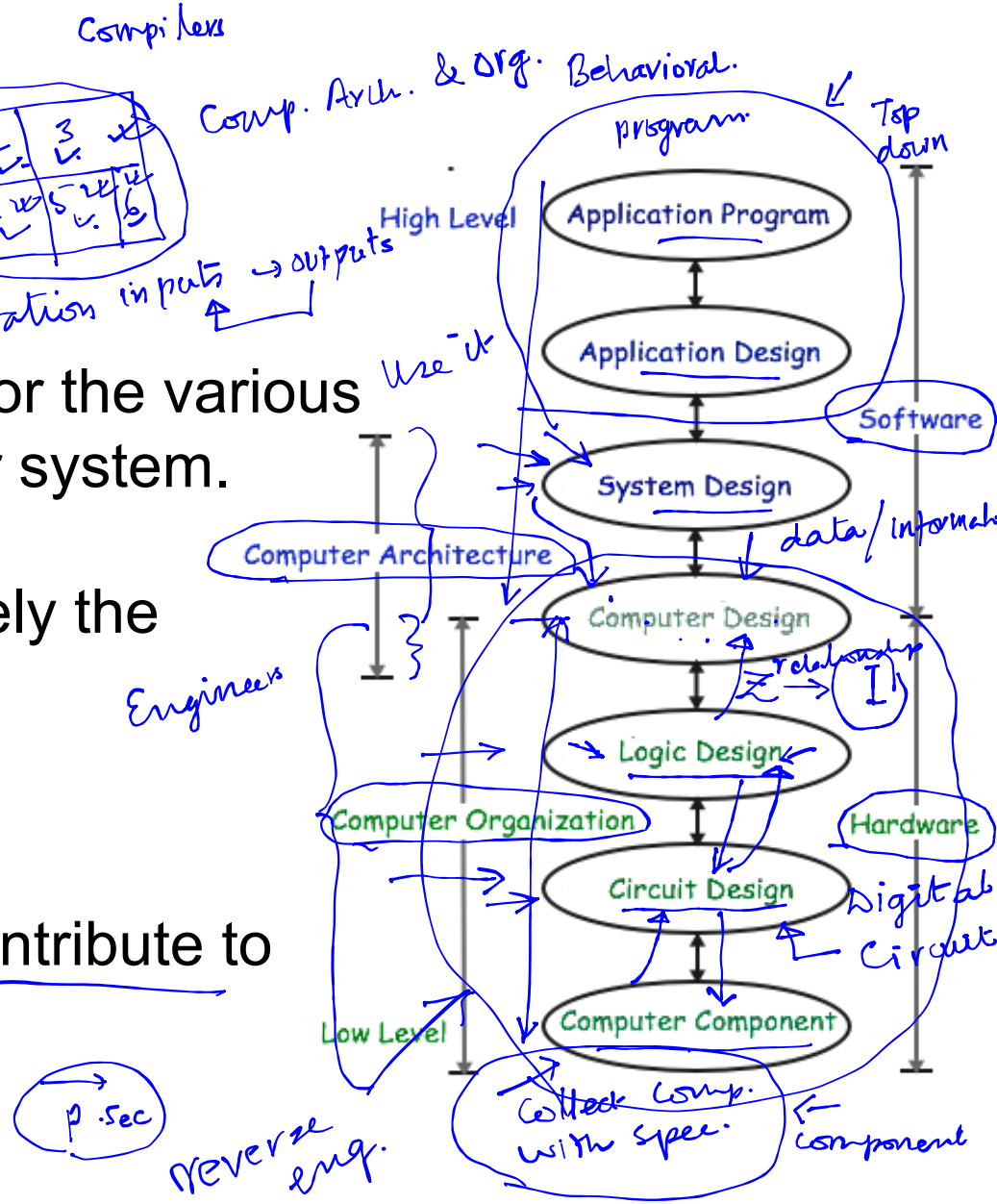
A functional description of requirements and design for the various parts of a computer: Functional behavior of computer system.

Architecture describes what the computer does: largely the instruction set

Computer Organization

How operational attributes are linked together and contribute to realize the architectural specification.

The structural relationship between components



Grades are important 😐
(only in students' and admin's mind)

- For branch change
- For scholarship/TA'ship during BTech
- For good internship
- For placements
- For BTP
- For good research project
- For recommendation letters
- For admission and scholarship for MS/PhD

In my opinion these are all a myth except:
Branch change, Scholarship/TA'ship during B.Tech and good internship and placement through Placement Cell

Designing Digital Systems: Where are We Going?

Where are *we* going?

- We're going to take a relatively short tour through the design of digital systems.
- When you are done with the course, we will be able to design some such systems.
- Examples could be a washing-machine timer or a vending-machine controller.
- Watch out!
  - You'll know just enough to be dangerous!
  - You are not going to get very deep into this design process—you can't in the time we have.
  - But you will know the basics and you can build on those.

What are we going to do?

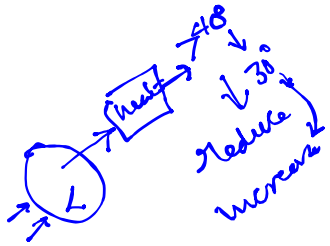
This course is divided into eight parts that pretty much cover the major aspects of digital systems and their design:

1. Introduction to designing digital systems with some examples, plus a description of some of the devices that we can use in our designs.
2. Number system and arithmetic - We do need to know a little about binary numbers and how they relate to decimal and perhaps hexadecimal, as well as some simple binary arithmetic.
3. Boolean algebra - This algebra is to digital systems what calculus is to mechanics. We really need just a few of the basic theorems and an understanding of how this all fits together.
4. Combinational logic - The kind of logic where time is not a factor and there is no memory. Binary signals are combined to produce an immediate output. All digital systems will have combinational circuits within them.

no memory

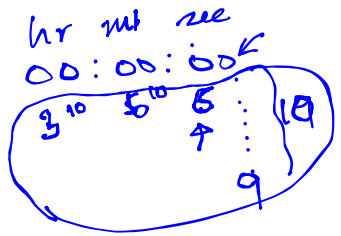
5. Building blocks combine into single chips many of the basic logic devices that we will use in designing combinational logic circuits. These blocks can be quite extensive, containing many circuit elements. These save us work when we are designing a system because they often will do on one chip what many basic chips could do.

6. Sequential Logic – It is where the fun is! These systems involve time and memory, remembering the past and acting upon both past events and current requests. These contain memory devices such as flip-flops. An example is the coin collector on a vending machine. You want it to record your 50% whether you put in Rs. 5, Rs. 2 or Rs. 1 or 50p. No combinational circuit can do that because it can't remember how much money has already been inserted.



7. **Registers and Counters** – These are particular instances of sequential circuits. They are so common in digital systems that there are standard building blocks that handle these functions. Counting the amount of money dropped into the coin slot of a vending machine is a good example of the use of a counter.

8. Once we have all of this learning behind us, we will be ready to design finite state machines (fsm), which even sounds hairy! There are some pretty well-organized design techniques for doing this.

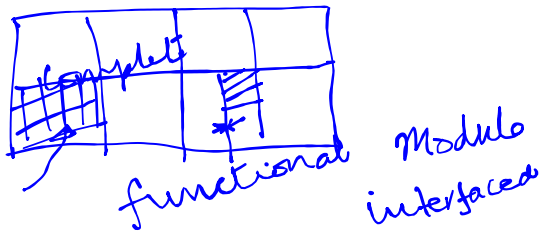


. . . . . .

→ clock
combinational lock

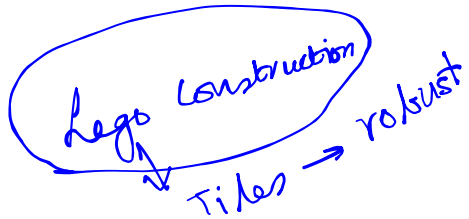
DESIGN—UP AND DOWN

- When you start working a problem, one that doesn't have a single obvious solution, how do you start?
- We are excluding problems, which has one answer and there are not very many ways of getting to it.
- Design problems have many solutions.
- It generally don't have simple, fixed paths along which the design proceeds.
- They often involve going in the wrong direction, making the wrong decision, backing up and starting over, and so on.
- They generally have no single answer but rather can be solved lots of different ways.



Top-down approach

- The top-down approach involves looking at the big picture and breaking that picture up into smaller constituents.
- Each of these smaller parts is a complete entity, which will combine with all these other parts to make the finished whole.
- But the smaller constituents may still be too large for one person or a small design team to handle.
- So we break them up again, getting even smaller constituents.
- Eventually, we'll have these constituents small enough that we can handle them, one by one, fairly easily.



Bottom-up approach

- If we are careful to break the problem at clear boundaries between subsystems, we will find that we have small constituents whose integrity we can be sure of.
- Then we can test each constituent completely, making certain that it does its job 100% correctly.
- When these are assembled into the larger system they were broken out from, the larger system should also work correctly.

DESIGN—UP AND DOWN

- How far must you go in breaking a problem down into these constituent parts?
- That depends (which probably isn't the answer you wanted to hear!).
- You break the system down until you have parts you can handle.
- You might perhaps choose to break the system into smaller parts than I would; maybe you would choose not to have so many smaller parts.
- It's really a judgment call based on how good you are at designing things, where the natural boundaries are, and what you and your team feel they can handle.

WHAT'S A DIGITAL SYSTEM?

- A digital system is one that involves discrete quantities, most often binary digits – *digits that are two valued viz. 0 and 1*
- What we need to know now is what kinds of *things* we will find in digital systems.
- The basic units are gates, which are devices that combine binary signals in fixed, simple ways.
- Let us now look at some of the basic operations (gates) that we have.

- AND ---- For AND operation on two variables, say *X* and *Y*, the operation results in 1 if and only if both the variables are 1 and the result is denoted by $F = X \cdot Y$, where F is the result of the operation. If there are more than two variables, then the operation results in 1 iff all the variables are 1, else it will be 0.
- OR ---- For OR operation on two variables, say *X* and *Y*, the operation results in 0 if and only if both the variables are 0 and the result is denoted by $F = X + Y$, where F is the result of the operation. If there are more than two variables, then the operation results in 0 iff all the variables are 0, else it will be 1.
- NOT ---- For operation on one variable only, say X. This operation results in 0 if the variable is 1 and results in 1 if the variable is 0 and the result is denoted by $F = \overline{X}$ or $F = X'$, where F is the result of the operation. This is also called complementing and F is called the logical complement of X.
- We can combine gates into fancier combinational circuits such as the multiplexer. This device allows us to steer any one of a number of inputs to a single output.