Introduction to Python

while (!(succeed = try()));

Kailash Prasad
PhD Scholar in Electrical Engineering
IIT Gandhinagar
Email: kailash.prasad@iitgn.ac.in
Website: https://constantnit.github.io/kailashprasad/
LinkedIn: https://www.linkedin.com/in/kailash-prasad/

About Me

- **B.Tech in Electronics and Communication Engineering**, NIT Arunachal Pradesh, 2014-2018
- **Ph.D Scholar in Electrical Engineering**, IIT Gandhinagar, 2018 - Present

Acknowledgement

Dr. Subhashish Banerjee

Assistant Professor ,Computer Science and Engineering

NIT Arunachal Pradesh

Shivaditya Katihar and His Team

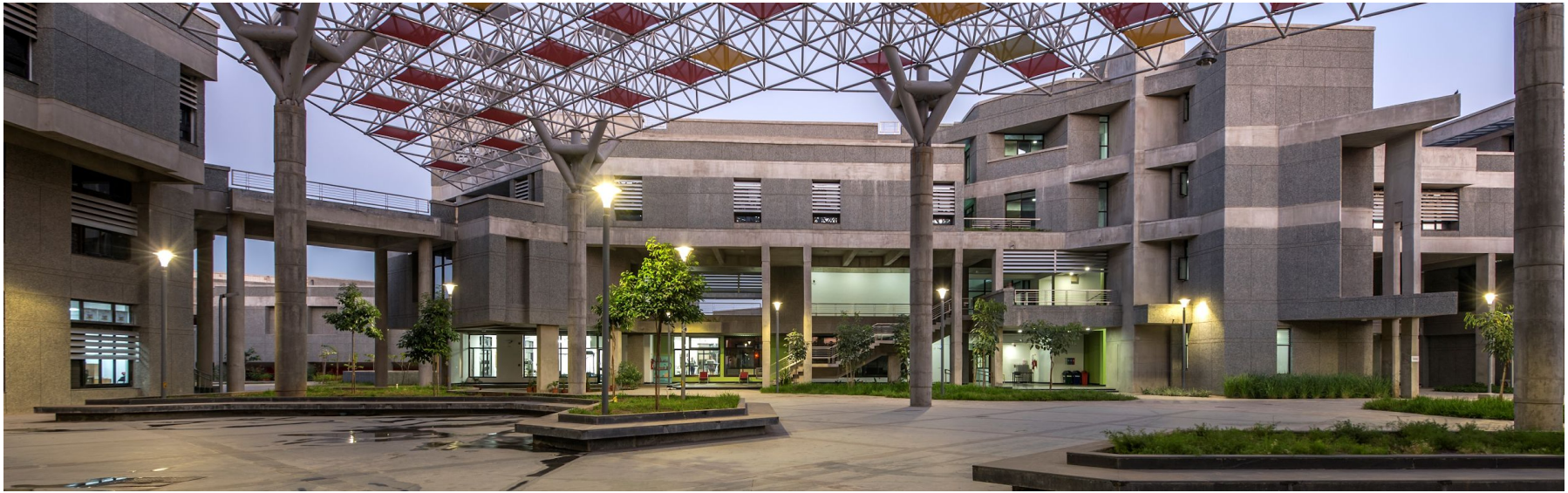
B.Tech NIT Arunachal Pradesh

Chandan Kumar Jha

PhD Electrical Engineering, IIT Gandhinagar

NIT Arunachal Pradesh

IIT Gandhinagar



http://events.iitgn.ac.in/2020/NCONC-2020/homepage.html

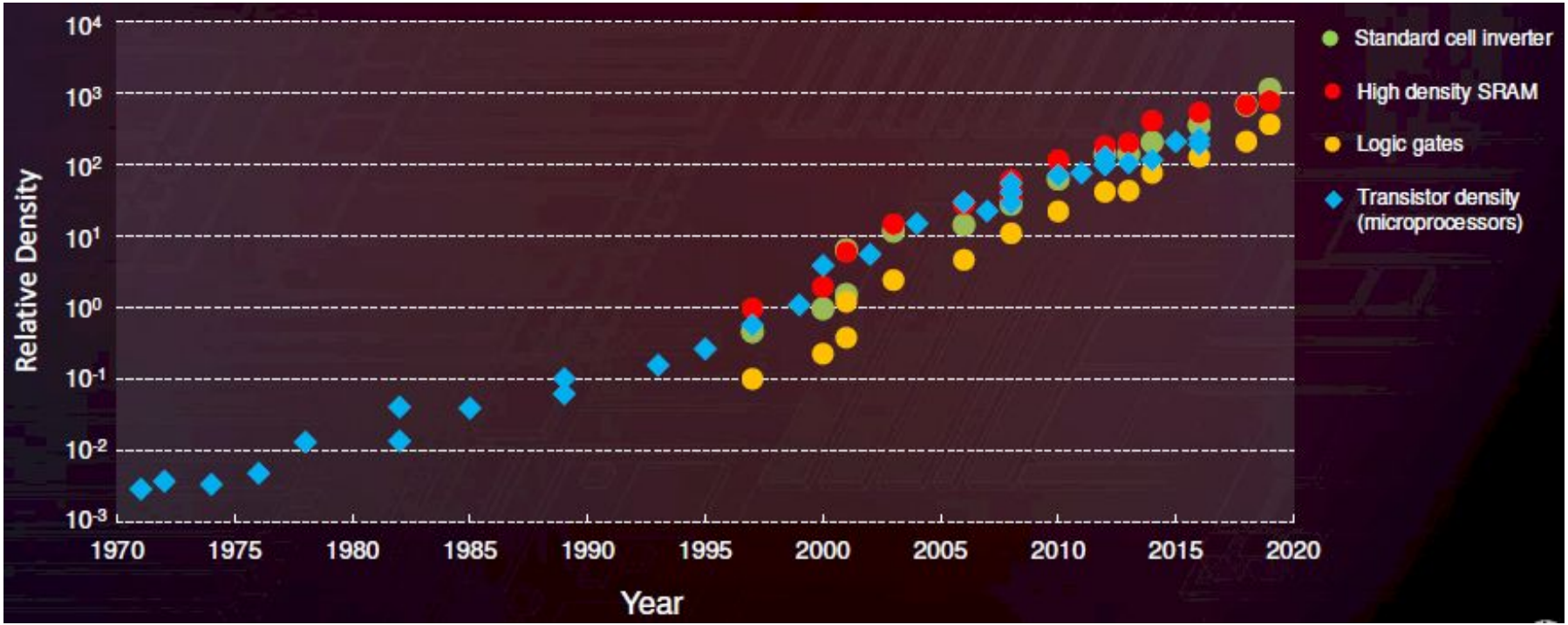
Goal

nanoDC should speak in one language!

Basic Needs of Life

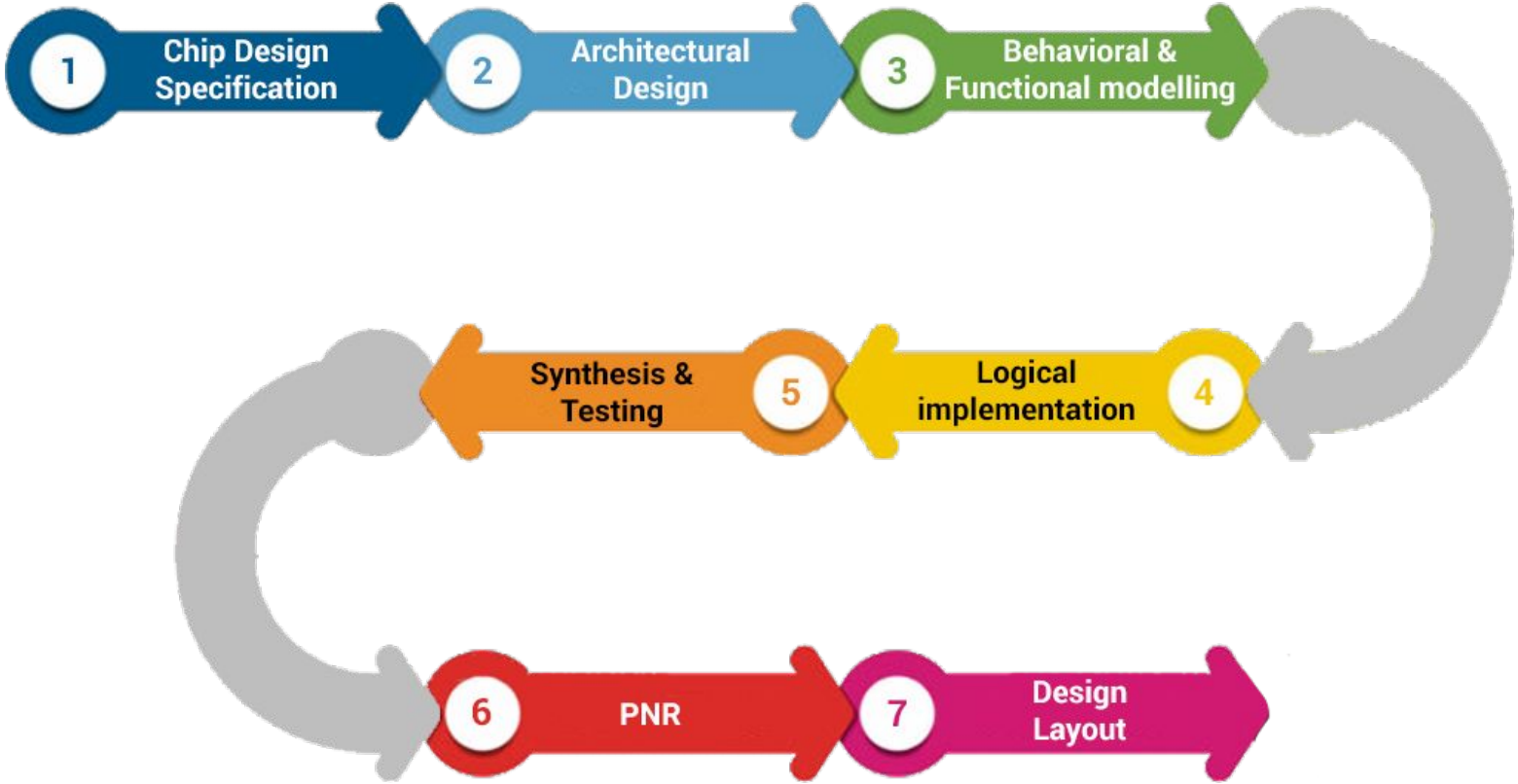
</>

Moore's Law



Source: TSMC

RTL to GDSII



Source: https://www.einfochips.com/blog/asic-design-flow-in-vlsi-engineering-services-a-quick-guide/

Python

Source:https://data-flair.training/blogs/features-of-python/

What you can do with python?

Source: https://data-flair.training/blogs/why-learn-python/

Programmers are more important than programs

import this

```
The Zen of Python, by Tim Peters
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Where to practice?

https://www.hackerrank.com/dashboard

Reference

- https://github.com/rajathkmp/Python-Lectures
- https://github.com/jupyter/jupyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks
- https://github.com/zhiyzuo/python-tutorial
- https://gist.github.com/kenjyco/69eeb503125035f21a9d
- https://github.com/mGalarnyk/Python_Tutorials
- https://github.com/jerry-git/learn-python3
- **https://www.tutorialspoint.com/python3/index.htm**

Day 1

- **Motivation**
- **Installing Jupyter Lab**
- Some Best Coding Practice
- Python Basics
  - Numbers
  - Strings
  - Conditionals

Day 2

- Python Basics
  - **Numbers**
  - **Strings**
  - **Conditionals**
  - **Loops**
  - **Lists**

Day 3

- Python Basics
  - **Numbers**
  - **Strings**
  - **Conditionals**
  - **Loops**
  - **Lists**
  - **Dictionaries**
  - **Functions**
  - **Modules**

Day 4

- Python Basics
  - **Numbers**
  - **Strings**
  - **Conditionals**
  - **Loops**
  - **Lists**
  - **Dictionaries**
  - **Functions**
  - **Modules**
  - **Exception Handling**
  - **A simple Project**

Day 5

- Python Basics
  - **Numbers**
  - **Strings**
  - **Conditionals**
  - **Loops**
  - **Lists**
  - **Dictionaries**
  - **Functions**
  - **Modules**
  - **Exception Handling**
  - **A simple Project**
  - **Standard Libraries**
    - **Datetime, logging, random, re**

Day 6

- Python Basics
  - **Numbers**
  - **Strings**
  - **Conditionals**
  - **Loops**
  - **Lists**
  - **Dictionaries**
  - **Functions**
  - **Modules**
  - **Exception Handling**
  - **A simple Project**
  - **Standard Libraries**
    - **Datetime, logging, random, re**
  - **Some examples**

Day 7

- Python Basics
- **Python Advanced**
  - **Numpy**