# Project 4 Review

Tawfik Osman

# Download files and Test Run

- To start this project
- You need to download this file: cmoslibrary.lib, “InvChain.sp” , “run_hspice.py”
- Copy file to the eecad machine and run the run_hspice.py with InvChain.sp as the input Hspice file.
- The subprocess.Popen() command in the run_hspice.py file runs the hspice file within the python environment. Then it generates output that is stored as output by this line of code “output, err = proc.communicate()”
- Now whenever the subprocess.Popen() is call, the command generates a csv file that contains the measured values determined by the hspice command **.measure** in the inputted hspice file “InvChain.sp”. The csv file is stored with the same name as the hspice file, thus in this case it is “InvChain.mt0.csv”.
- Now what you should note is that, you can access the result from running the hspice file from either the output or the csv file. And what we need from the running the hspice file is the "tphl_inv“ measured by the **.measure** command in the “InvChain.sp” file. We can get this by using regular expression to sparse through the output file and pick out the line having “tphl_inv” or we just read the csv file and pick out the values that correspond to the header “tphl_inv”.
- Now if you run the “run_hspice.py” file for the first time with “InvChain.sp” as the input hspice desk to the subprocess.Popen() command , the "tphl_inv” value printed from csv file is the time delay for “InvChain.sp” with **11 inverters** and a **fan of 7.**

# Project Expectations

- Now lets assume you have been able to run the “run_hspice.py” in the eecad machine, the next step is to create function that takes inputs of “N”, “fan”, “filename” and generates an hspice file with the same filename. For example, you can approach this, by

def(N,fan,filename):

open the filename in a write mode “n=open(“filename.sp”, “w”)

Write all the unchangeable parameters into this filename.sp ( line1 =“Lab 1 Problem 1A” , then n.write(line1) ). You can each of the unchangeable line to the file

Following the instruction given in class you can make **.measure** line of code unchangeable.

You change the **.param fan** = # for every fan given

Update the number of inverter with **the N and fan** given

Xinv1 a b inv M=1

Xinv2 b c inv M=fan**1

Xinv3 c d inv M=fan**2

…

…

**NB you can ask question in my office hours if you are not cleared**

- For those who wants to use regular expression, you can create another function that takes ouput of subprocess.Popen() as it input. In the function, you can read each line in the output, decode it, and use re.search to look for “tphl_inv”, if you find it you strip() and split() it to make it accessible by indexing. Then you can you indexing to pick out the variable “tphl_inv” and its corresponding value.
- For simplicity, using the csv file option is easier, thus after generating your hspice file, the next thing is to run it with subprocess.Popen() , which means the “InvChain.mt0.csv” will automatically be generated, then you read the value of “tphl_inv”using the approach given in “run_hspice.py” file given

# Finding the best N and fan value

- Now if you can successfully generated hspice file and read the “tphl_inv” value from it after running it on the eecad machine for any given N and fan value. The next thing is to find the N and fan value that will give us the smallest time delay.
- You can create a list of N and fan separately. Note N= [2,3,4,5,6….] and fan =[1,3,5,7,9…] . Fan should be odd integers.
- Now you can initialize a larger N and fan value as advise by Professor.
- You can use a nested loop of N and fan. for each value of N and fan, you input them to your hspice generator function, then the output in taken as an hspice input to the subprocess.Popen() command. The “tphl_inv” is read and compared with the initialized value, if less than, then you update your “tphl_inv” . Finally you store the N and fan with less time in a new set of variables.

NB: remember **to print the N, fan and tphl_inv** for each combination of your N and fan.

- You can now write a very short function that takes the N_min ,fan_min, filename. Use them as an input to the hspice generator function, generate and run the hspice and finally read the “tphl_inv” corresponding to those input. This function will just be similar to run_hspice.py with the inclusion of hspice generator function for the generating the hspice.