Task-1 Create GitHub repo. Document summary of previous video in the GitHub repo

Summary:
The First lecture was about Digital VLSI SoC Design and planning

In this lecture, we were taught about the initial overview of how applications run on hardware and how the hardware is planned to be built . Initially, we were told about the GCC Compiler which runs on linux , where the C code of the application is written and tested, the result of which is lets say O0

First step is to create an architecture specification in which the application runs on , in C whose output is O1 , then we refine this spec, making sure O0=O1. After which we make a softcopy of the hardware taking into consideration the specs which we made the previous code in Verilog, lets say the output of this is O2. Then starts the Soc Design step where we process this RTL code separately into 2 blocks:- Processors and Peripherals/IPs . The Processors will be written in a synthesizable code bringing it down to Gate Level Netlist . The Peripherals are divided into 2 parts :- the Macros, which are the some of the blocks which are repeated multiple times (clock dividers etc) and Analog IPs , which are used to interact with the analog signal (ADCs and DACs) . The processors and Macros from Peripherals can be synthesized using RTL where as the Analog IPs are made using MOSFETs . The final stage is that of SoC Integration, where these components are brought together into a single chip. The final output of the Soc which lets call as O3 is made sure to be equal to O2,O1,O0. Hence, making it a suitable hardware to carry out the application . Then we talk about the GDSII and the how big it is , the checks that we do for these files and the tape-out, tape-in process, etc Then we discuss the difference between microprocessors and microcontrollers . After the final chip in taped-in we check the output O4 and check whether it is same as other outputs .

Task-2 Install tools listed in this document using the machine configuration mentioned. Update

your GitHub repo with Tool snapshot

----—--Installation instructions ---

Oracle virtual machine link

https://www.virtualbox.org/wiki/Downloads

System Check

6GB RAM, 50 GB HDD

Ubuntu 20.04+

4vCPU

Tool check

Yosys

$ sudo apt-get update

$ git clone https://github.com/YosysHQ/yosys.git

$ cd yosys

$ sudo apt install make (If make is not installed please install it)

$ sudo apt-get install build-essential clang bison flex \

 libreadline-dev gawk tcl-dev libffi-dev git \

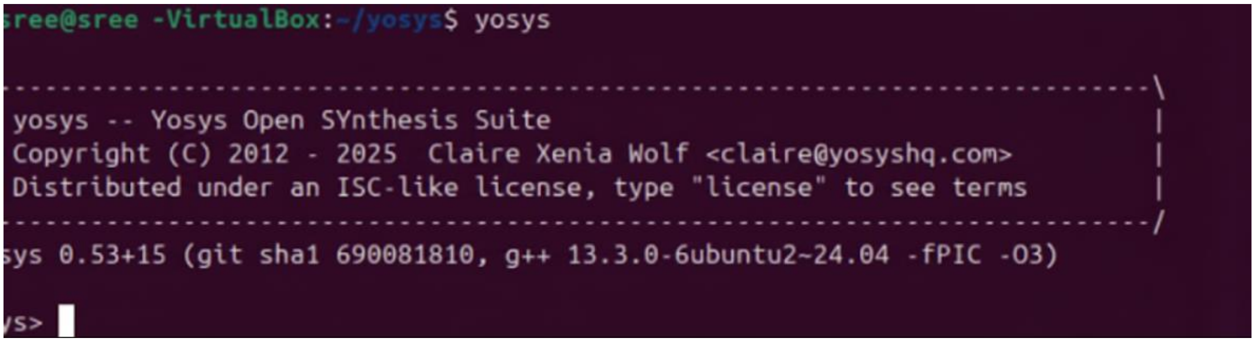
 graphviz xdot pkg-config python3 libboost-system-dev \

 libboost-python-dev libboost-filesystem-dev zlib1g-dev

$ make config-gcc

$ make

$ sudo make install



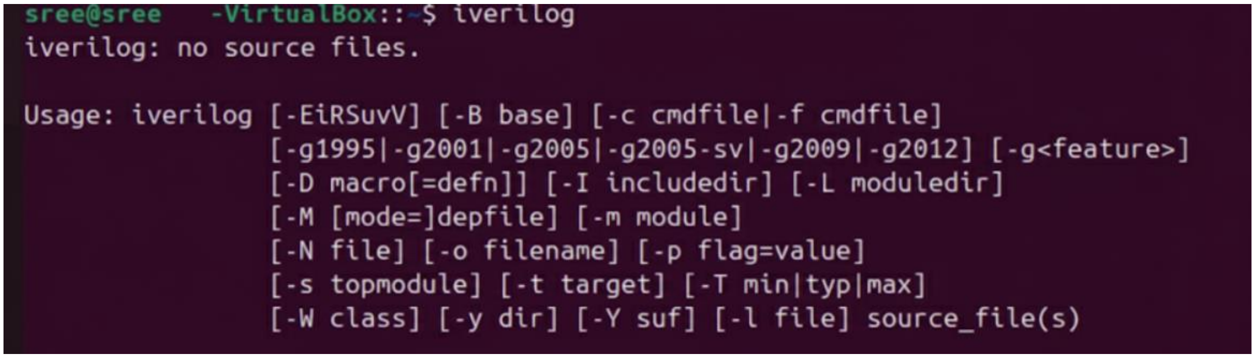
```
ree@sree -VirtualBox:~/yosys$ yosys

---------------------------------------------------------------------------\
 yosys -- Yosys Open SYnthesis Suite                                        |
 Copyright (C) 2012 - 2025  Claire Xenia Wolf <claire@yosyshq.com>         |
 Distributed under an ISC-like license, type "license" to see terms       |
---------------------------------------------------------------------------/
sys 0.53+15 (git sha1 690081810, g++ 13.3.0-6ubuntu2~24.04 -fPIC -O3)

s> 
```

Iverilog

Steps to install iverilog

sudo apt-get update

sudo apt-get install iverilog



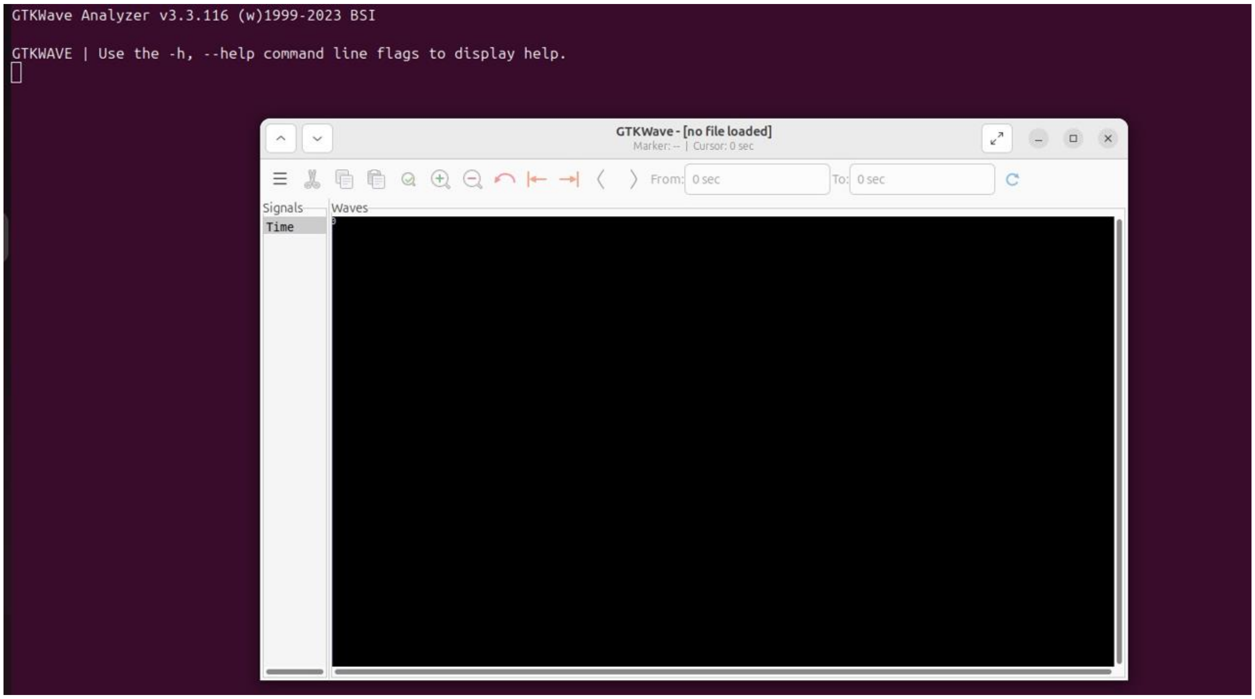
```
sree@sree   -VirtualBox::~$ iverilog
iverilog: no source files.

Usage: iverilog [-EiRSuvV] [-B base] [-c cmdfile|-f cmdfile]
                [-g1995|-g2001|-g2005|-g2005-sv|-g2009|-g2012] [-g<feature>]
                [-D macro[=defn]] [-I includedir] [-L moduledir]
                [-M [mode=]depfile] [-m module]
                [-N file] [-o filename] [-p flag=value]
                [-s topmodule] [-t target] [-T min|typ|max]
                [-W class] [-y dir] [-Y suf] [-l file] source_file(s)
```

gtkwave

Steps to install gtkwave

sudo apt-get update

sudo apt install gtkwave



```
GTKWave Analyzer v3.3.116 (w)1999-2023 BSI

GTKWAVE | Use the -h, --help command line flags to display help.
```