CROSS COMPILATION

Note:-

1. For Installing KVM and Network bridge configuration, please connect ethernet LAN CAT 5 cable to your laptop for internet access. Internet will only work via ethernet cable in Virtual Machine.
2. We have retained the source code and modified or included makefile to run the code on different platforms.

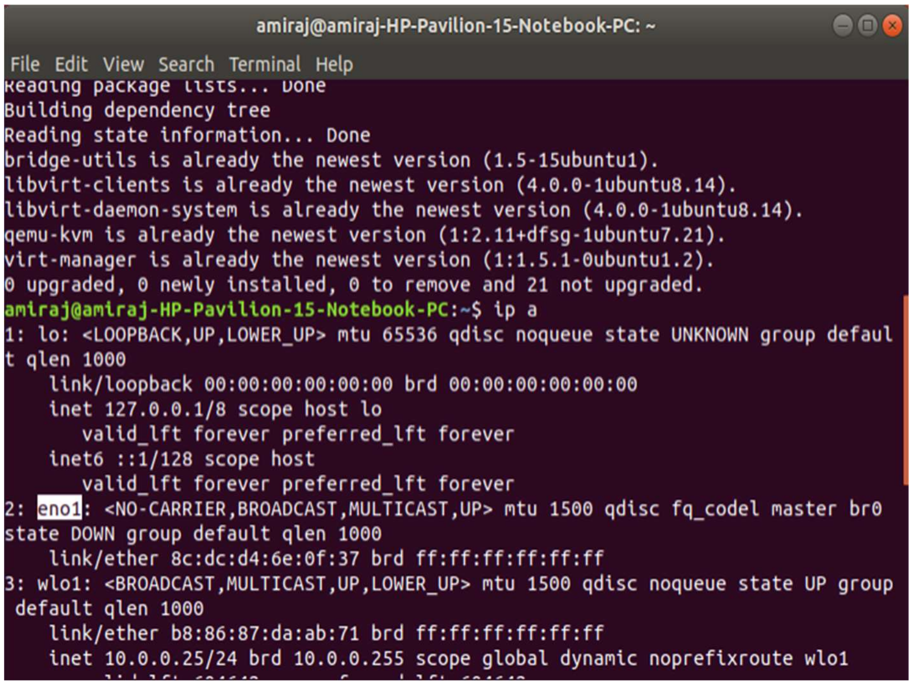
Reference for code-
POLAR CODE: https://github.com/tavildar/Polar
TURBO CODE: https://github.com/Poo19/Turbo-Encoder-Decoder
OPERATING SYSTEM: LINUX 18.04 LTS

1. Update package version and their dependencies, open terminal : **"sudo-apt get update"**

2. Run command(to install package)- **"sudo-apt-get install qemu-kvm libvirt-clients libvirt-daemon-system bridge-utils virt-manager"**

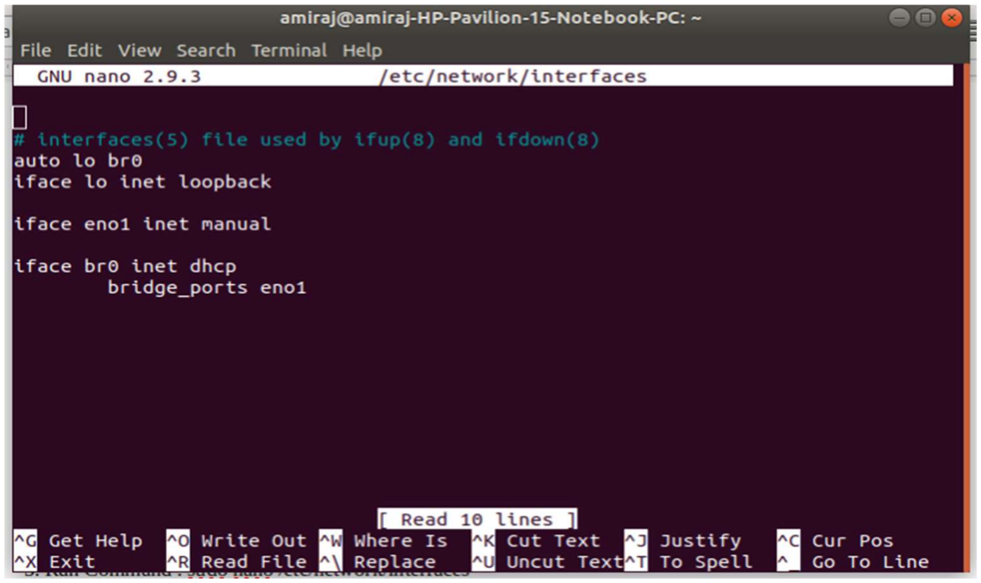
3. Need to set up network bridge-Bridging allows our virtual machine to access the network.

4. Run command: **"ip a"**; note down the onboard network interface card address of system as per the below image.


amiraj@amiraj-HP-Pavilion-15-Notebook-PC: ~
File Edit View Search Terminal Help

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
bridge-utils is already the newest version (1.5-15ubuntu1).
libvirt-clients is already the newest version (4.0.0-1ubuntu8.14).
libvirt-daemon-system is already the newest version (4.0.0-1ubuntu8.14).
qemu-kvm is already the newest version (1:2.11+dfsg-1ubuntu7.21).
virt-manager is already the newest version (1:1.5.1-0ubuntu1.2).
0 upgraded, 0 newly installed, 0 to remove and 21 not upgraded.
amiraj@amiraj-HP-Pavilion-15-Notebook-PC:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel master br0
state DOWN group default qlen 1000
    link/ether 8c:dc:d4:6e:0f:37 brd ff:ff:ff:ff:ff:ff
3: wlo1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group
 default qlen 1000
    link/ether b8:86:87:da:ab:71 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.25/24 brd 10.0.0.255 scope global dynamic noprefixroute wlo1
```

5. Run Command : **“sudo nano /etc/network/interfaces”**
Now, mention details as per below image.

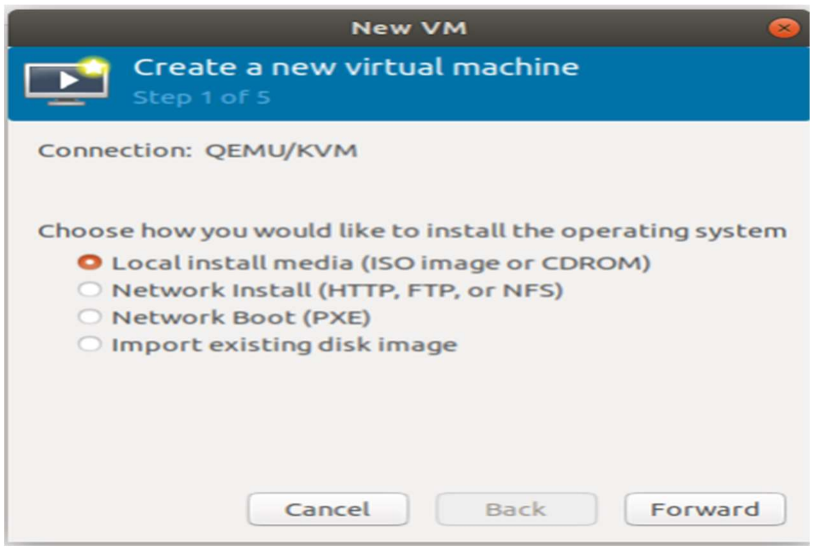

amiraj@amiraj-HP-Pavilion-15-Notebook-PC: ~
File Edit View Search Terminal Help

```
  GNU nano 2.9.3              /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo br0
iface lo inet loopback

iface eno1 inet manual

iface br0 inet dhcp
        bridge_ports eno1







                          [ Read 10 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

Now, Save the bridge network protocol in etc/network directory(Ctrl+X, Y)

6. Now, Virtual Machine Manager window will pop up, select file then Create new Virtual Machine

7. Chose local install media as per below image



Connection: QEMU/KVM

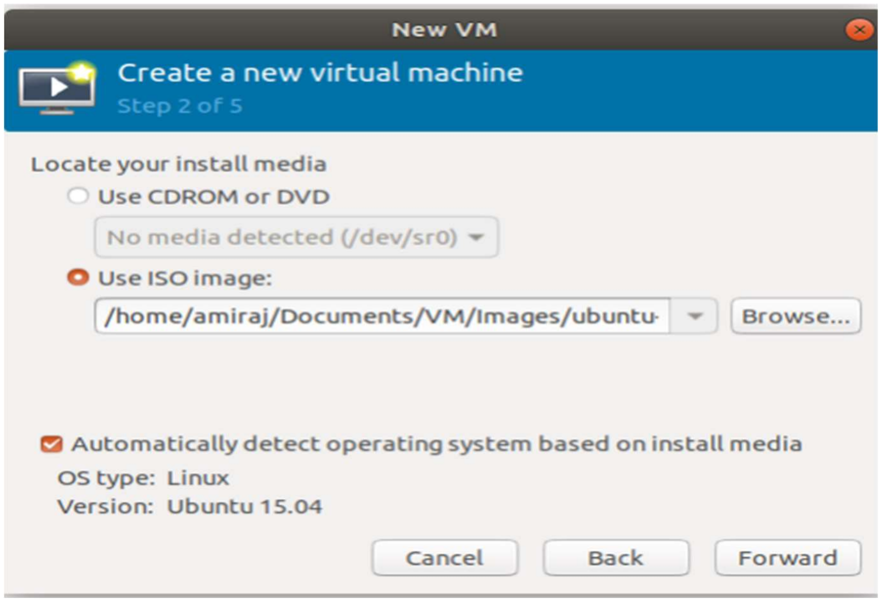
Choose how you would like to install the operating system

- [✓ Local install media (ISO image or CDROM)]
- Network Install (HTTP, FTP, or NFS)
- Network Boot (PXE)
- Import existing disk image

Note:- It will by default select x86 architecture.7. Meanwhile, download Ubuntu 15.04 64-bit PC (AMD64) desktop image from http://old-releases.ubuntu.com/releases/15.04/

8. Select ubuntu-15.04-desktop-amd64.iso as per below image.

Note:- KVM will automatically detect OS type and Version(it may take some time to process)



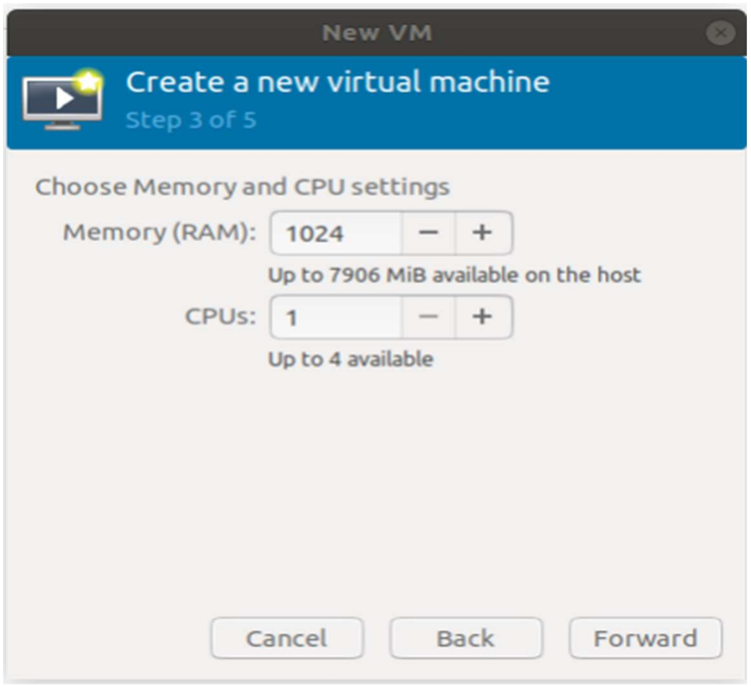
Locate your install media

- Use CDROM or DVD
  No media detected (/dev/sr0)
- [✓ Use ISO image:]
  /home/amiraj/Documents/VM/Images/ubuntu

[✓] Automatically detect operating system based on install media

OS type: Linux
Version: Ubuntu 15.04

9. Now, Select Memory and CPU as per below image.



Choose Memory and CPU settings

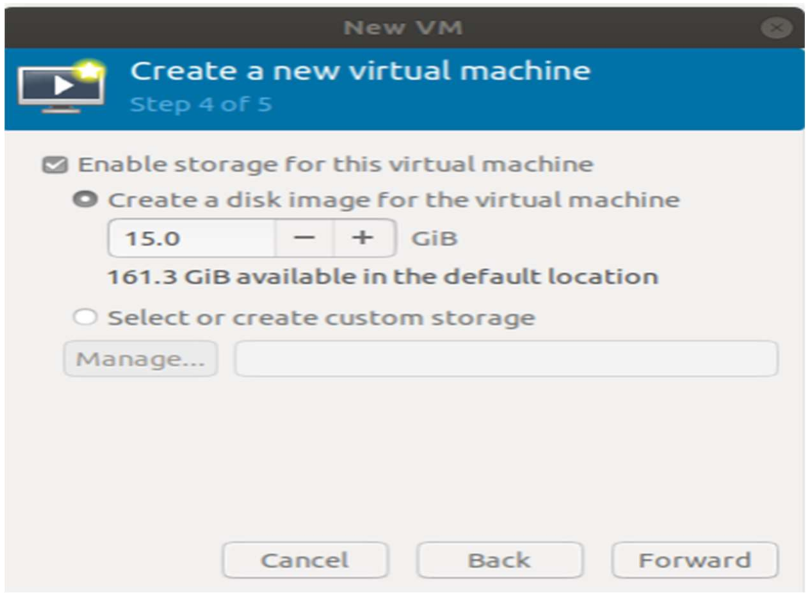
Memory (RAM): 1024

Up to 7906 MiB available on the host

CPUs: 1

Up to 4 available

10. Now, provide 15 GB for Virtual machine storage.



☑ Enable storage for this virtual machine

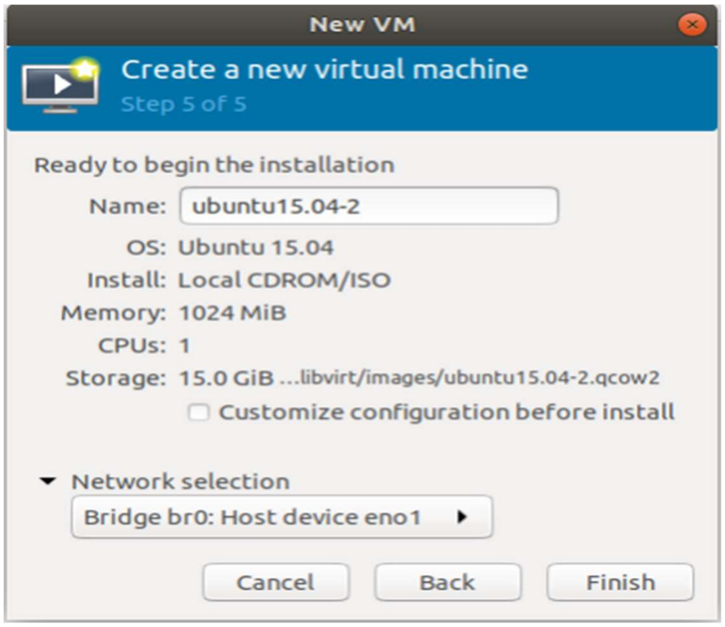
● Create a disk image for the virtual machine

15.0 GiB

161.3 GiB available in the default location

○ Select or create custom storage

11. Select Bridge br0 in Network Selection.



Ready to begin the installation

Name: ubuntu15.04-2

OS: Ubuntu 15.04

Install: Local CDROM/ISO

Memory: 1024 MiB

CPUs: 1

Storage: 15.0 GiB ...libvirt/images/ubuntu15.04-2.qcow2

☐ Customize configuration before install

Network selection

Bridge br0: Host device eno1

12. Now, enter the command in terminal: **“sudo service networking restart”**

13. Return to KVM, and press finish.

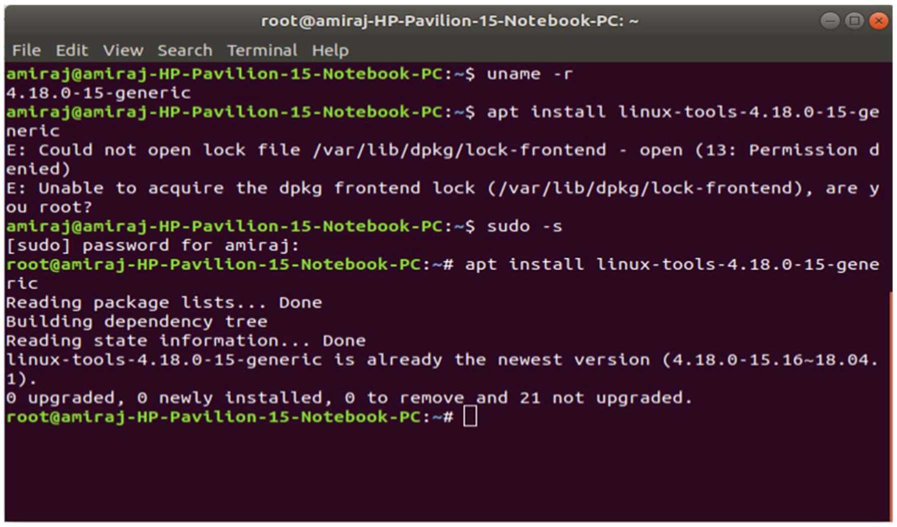
14. Ubuntu will start to install in Virtual Machine. Follow the normal steps followed to install ubuntu.

16. Till here we have successfully installed Ubuntu 15.05 in KVM on x86 architecture. Now, we will run Polar code and Turbo code and check performance on our host computer and then on guest computer.

17. Now, we will install packages for perf stat command which used to calculate the performance of CPU, which is as follows in below steps

18. Find the version of your kernel, use command: **“uname -r”**

19. Enter command to install packages: **“apt install linux-tools-kernelname”**(refer below reference image)



root@amiraj-HP-Pavilion-15-Notebook-PC: ~
File Edit View Search Terminal Help

```
amiraj@amiraj-HP-Pavilion-15-Notebook-PC:~$ uname -r
4.18.0-15-generic
amiraj@amiraj-HP-Pavilion-15-Notebook-PC:~$ apt install linux-tools-4.18.0-15-ge
neric
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission d
enied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are y
ou root?
amiraj@amiraj-HP-Pavilion-15-Notebook-PC:~$ sudo -s
[sudo] password for amiraj:
root@amiraj-HP-Pavilion-15-Notebook-PC:~# apt install linux-tools-4.18.0-15-gene
ric
Reading package lists... Done
Building dependency tree
Reading state information... Done
linux-tools-4.18.0-15-generic is already the newest version (4.18.0-15.16~18.04.
1).
0 upgraded, 0 newly installed, 0 to remove and 21 not upgraded.
root@amiraj-HP-Pavilion-15-Notebook-PC:~#
```

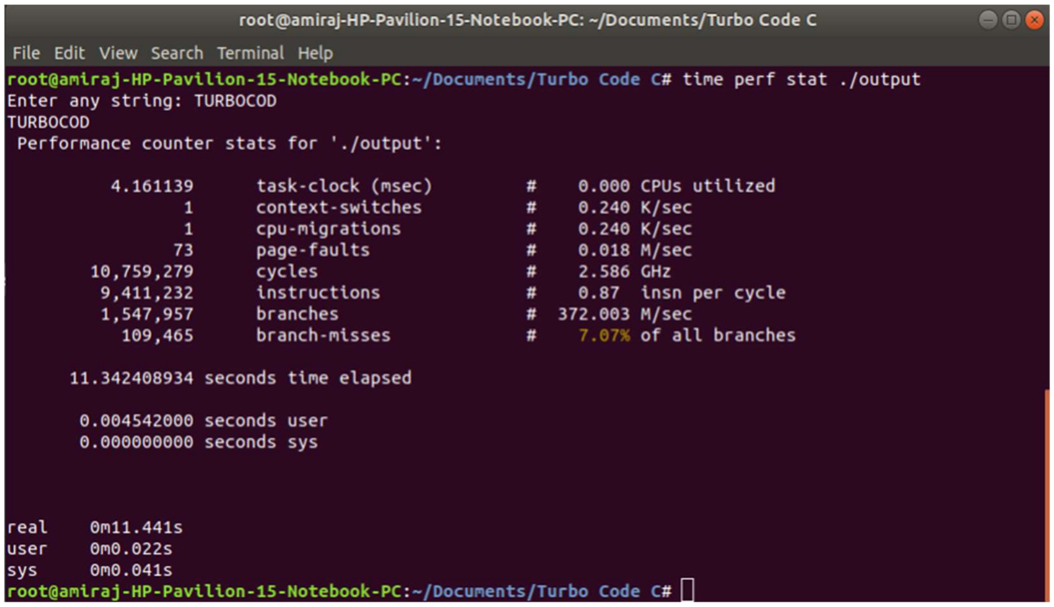
Note:- You may get error of permission denied and dpkg frontend lock, write **"sudo – s"** command to enable root. After this package will install.

20. Enter command: **"perf –help"** (it will display usage of perf command and most used commands)

21. Now, open Turbo Code C folder and open in terminal in the same folder.

Note:- We have generated make file for Turbo_Codes.c Deconcatination.c Deinterleaver.c Encoder.c Interleaver.c Max_Function.c Modulating.c Turbo_Testing_code.c boxmuller_noise.c, we need to only run output file to execute Turbo Code.

22. Now, enter the command: **"time perf stat ./output"**
and enter 64 bit string : TURBOCOD
We will get output like this as mentioned below:


root@amiraj-HP-Pavilion-15-Notebook-PC: ~/Documents/Turbo Code C
File Edit View Search Terminal Help

```
root@amiraj-HP-Pavilion-15-Notebook-PC:~/Documents/Turbo Code C# time perf stat ./output
Enter any string: TURBOCOD
TURBOCOD
 Performance counter stats for './output':

          4.161139      task-clock (msec)         #    0.000 CPUs utilized
                 1      context-switches          #    0.240 K/sec
                 1      cpu-migrations            #    0.240 K/sec
                73      page-faults               #    0.018 M/sec
        10,759,279      cycles                    #    2.586 GHz
         9,411,232      instructions              #    0.87  insn per cycle
         1,547,957      branches                  #  372.003 M/sec
           109,465      branch-misses             #    7.07% of all branches

      11.342408934 seconds time elapsed

       0.004542000 seconds user
       0.000000000 seconds sys


real	0m11.441s
user	0m0.022s
sys	0m0.041s
root@amiraj-HP-Pavilion-15-Notebook-PC:~/Documents/Turbo Code C# 
```

Now, we will enter strings of different bits as per the below mentioned table.

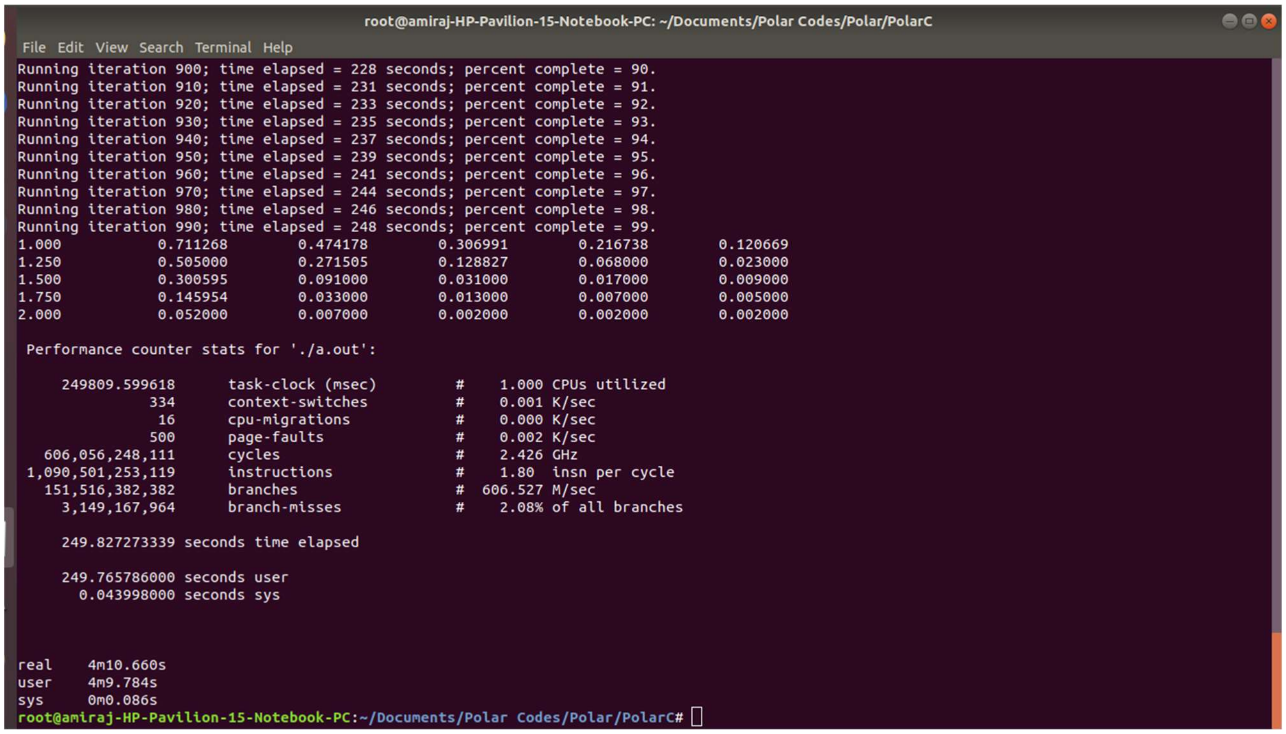
INFORMATION BITS (POLAR VALUES)	**STRING (TURBO VALUES)**
8	T
64	TURBOCOD
256	TURBOCODESAREACLASSOFFORWARDER
1024	Turbo codes are a class of forward error correction codes that provide a high performance in error correction, close to the chan
4096	Turbo codes are a class of forward error correction codes that provide a high performance in error correction, close to the channel capacity. The information sequence is encoded twice, with an interleaver between the two encoders serving to make the two encoded data sequences approximately statistically independent of each other.Often half rate Recursive Systematic Convolutional (RSC) encoders are used, with each RSC en-coder producing a systematic output which is equivalent to the original information sequ

512	Turbo codes are a class of forward error correction codes that p
128	Turbo codes area

23. Now, we will run polar code. Open the Polar C code. We have already generated the make file for main.cpp, PolarCode.cpp, Polarcode.h, PolarCode.gch

24. Now, open main.cpp file and change value of N=8, for encoding and decoding 8 bit of string.

25. Now, open in terminal in Turbo C folder and run command: **“time perf stat ./a.out”**



```
root@amiraj-HP-Pavilion-15-Notebook-PC: ~/Documents/Polar Codes/Polar/PolarC
File  Edit  View  Search  Terminal  Help
Running iteration 900; time elapsed = 228 seconds; percent complete = 90.
Running iteration 910; time elapsed = 231 seconds; percent complete = 91.
Running iteration 920; time elapsed = 233 seconds; percent complete = 92.
Running iteration 930; time elapsed = 235 seconds; percent complete = 93.
Running iteration 940; time elapsed = 237 seconds; percent complete = 94.
Running iteration 950; time elapsed = 239 seconds; percent complete = 95.
Running iteration 960; time elapsed = 241 seconds; percent complete = 96.
Running iteration 970; time elapsed = 244 seconds; percent complete = 97.
Running iteration 980; time elapsed = 246 seconds; percent complete = 98.
Running iteration 990; time elapsed = 248 seconds; percent complete = 99.
1.000           0.711268        0.474178        0.306991        0.216738        0.120669
1.250           0.505000        0.271505        0.128827        0.068000        0.023000
1.500           0.300595        0.091000        0.031000        0.017000        0.009000
1.750           0.145954        0.033000        0.013000        0.007000        0.005000
2.000           0.052000        0.007000        0.002000        0.002000        0.002000

 Performance counter stats for './a.out':

     249809.599618      task-clock (msec)         #    1.000 CPUs utilized
               334      context-switches          #    0.001 K/sec
                16      cpu-migrations            #    0.000 K/sec
               500      page-faults               #    0.002 K/sec
   606,056,248,111      cycles                    #    2.426 GHz
 1,090,501,253,119      instructions              #    1.80  insn per cycle
   151,516,382,382      branches                  #  606.527 M/sec
     3,149,167,964      branch-misses             #    2.08% of all branches

     249.827273339 seconds time elapsed

     249.765786000 seconds user
       0.043998000 seconds sys



real    4m10.660s
user    4m9.784s
sys     0m0.086s
root@amiraj-HP-Pavilion-15-Notebook-PC:~/Documents/Polar Codes/Polar/PolarC#
```

Note:- If you get an error : You may not have permission to collect stats.
Enter command **“sudo -s”** then run again.

26. Now, check value of each bits as per table mentioned in Sr. No 22 and check all the values of N.

26. Now, we have to repeat the same procedure on our guest machine i.e KVM Ubuntu 15.04 based on x86 architecture.

Note: - We need to transfer Polar code and Turbo Code folder from our host to guest machine. Copy all data in pen drive from host machine then open the running VM and click on virtual machine and redirect USB. Now, this will display USB drive in Virtual machine.