**Report on the D2500 ATOM Board:**

Testing the D2500 ATOM Board:

- **Result** :
  The board is working fine.

- The following are the preliminary steps to be carried out before testing.

  1. Physically disconnect the Hard Disk before booting the system.
  2.In the BIOS menu ,please make the following changes:
     - Disable the SATA controller.
     - Set the number of cores to 1
     - Check if serial ports are enabled.
     - Enable USB legacy ( This can be found in the Onboard devices menu).

- Steps to boot up and test the machine:
  - Create a bootable USB using steps mentioned in [1] and use it to start the machine.
  - When the grub menu appears. Please type the following commands : *

    ```
    -> linux16 /memdisk
    ->initrd16 /floppy.img
    ->boot
    ```

  - The system boots up and you will see a message saying "Loading Boot Sector ….Booting". The message indicates the system is ready for use.

* --This is a temporary testing step. An alternative way is to create a grub.cfg/ menu.lst file and place it in the boot/grub/ location in the USB stick. You would need to create a grub.cfg file if the bootable USB was created on a machine that runs the latest versions of Linux (Ubuntu 9.10 and higher) , for lower distros, please create a menu.lst file. (The download zip file from [1] details how to create a menu.lst file ).

**Testing**
The board was tested by running the programs taken from [1] ( Lab exercises 3-5). A bootable USB was created using the steps mentioned in the above website. The procedure to run the

programs are described below :

## Steps to compile and run a protected mode assembly program on a gdb setup

Files needed:

1. **gdbkama_prot** (script)
2. **prot_test.ld**
3. **prot_start.s** - Fit your code into this template.

Instructions:

1. You need "root" permission on the Linux host machine to execute this.
2. Copy the above two files into the same directory as the assembly code
3. Run script gdbkamap with filename as parameter (without the ".s" extension)
4. Run gdb
5. Give the following commands inside gdb:
    - ○ set remotebaud 38400
    - ○ target remote /dev/ttyS0
    - ○ load [filename].out
    - ○ c

Thus far,

- the assembly program has been compiled and an executable has been created (by script gdbkamap, using test.ld)
- program has been loaded into the target system and is currently executing

Now, using various gdb commands, we can check the values at specific memory locations, register values etc.

An Example:

Say the assembly program is myprog.s

```
$ ./gdbkamap myprog
$ gdb
(gdb) set debug remote 1
(gdb) set remotebaud 38400
(gdb) target remote /dev/ttyS0
(gdb) load myprog.out
(gdb) c
```

NOTE: To run in real mode, follow the same procedure as above, but with the following files:

1. **gdbkama_real**
2. **prot_real.ld**
3. real_start.s

**Errors & Warnings:**

The following errors and warnings were encountered during the simulation.

Error:

***Ignoring packet error, continuing...**
**warning: unrecognized item "timeout" in "qSupported" response**
WHEN:target remote /dev/ttyS0
SOLUTION: target system not reachable. Check system, serial cable

Also check if the serial port is connected to COM0 of the ATOM board.
The COM0 was not external in the D2500 board.

***Remote faulure reply: E01**
WHEN: load
SOLUTION: retry

***SIGSEGV**
WHEN: anytime
SOLUTION:retry

* INT 13 08 failure, assuming this is the only drive.
When: during boot
It is a warning message.

References:

[1].http://rise.cse.iitm.ac.in/people/faculty/kama/prof/Comp-Arch.html
[2]. A tutorial on gdb :http://www.dirac.org/linux/gdb/index.php
[3]Intel x86 Architecture manuals :http://rise.cse.iitm.ac.in/people/faculty/kama/prof/refs.html