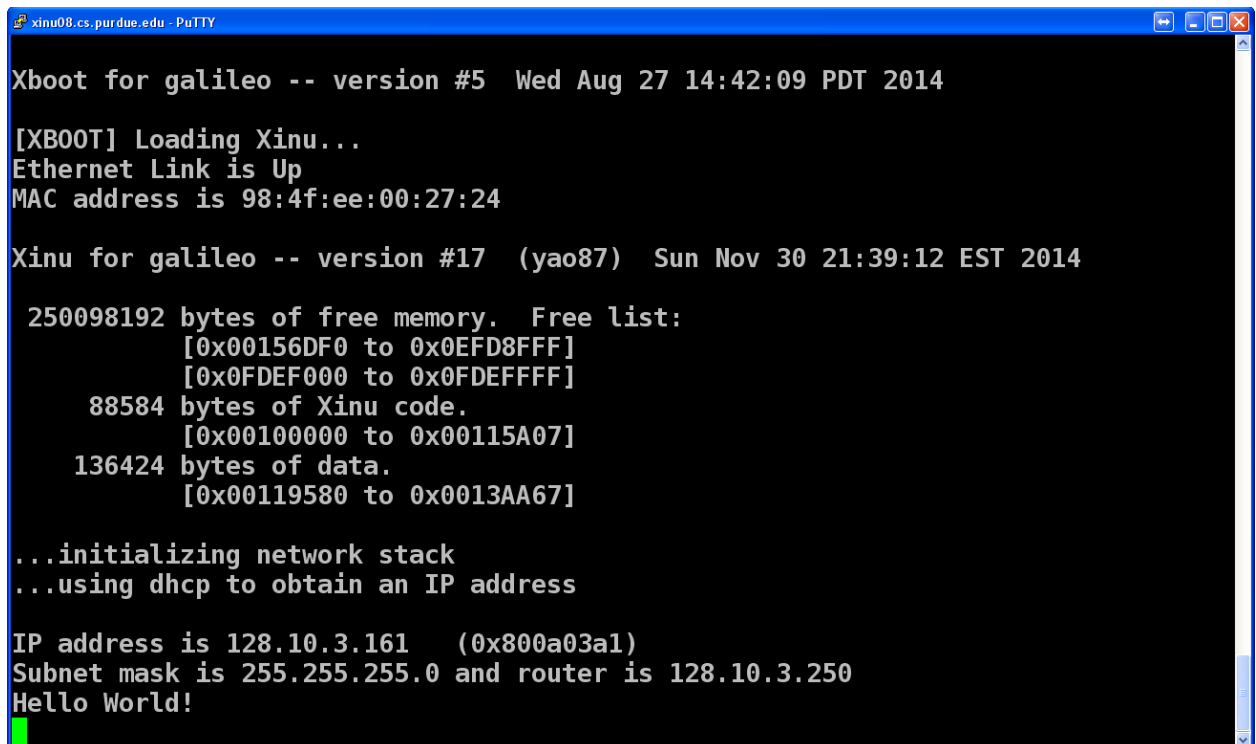


There are 5 switches for tests in main.c:

1. Shell test
2. Helloworld test
3. Hash table test
4. Load_library test
5. Load_library test for too many functions and duplicate functions

Test Result:

1. Helloworld



```
xinu08.cs.purdue.edu - PuTTY

Xboot for galileo -- version #5  Wed Aug 27 14:42:09 PDT 2014

[XBOOT] Loading Xinu...
Ethernet Link is Up
MAC address is 98:4f:ee:00:27:24

Xinu for galileo -- version #17  (yao87)  Sun Nov 30 21:39:12 EST 2014

 250098192 bytes of free memory.  Free list:
      [0x00156DF0 to 0x0EFD8FFF]
      [0x0FDEF000 to 0x0FDEFFFF]
   88584 bytes of Xinu code.
      [0x00100000 to 0x00115A07]
  136424 bytes of data.
      [0x00119580 to 0x0013AA67]

...initializing network stack
...using dhcp to obtain an IP address

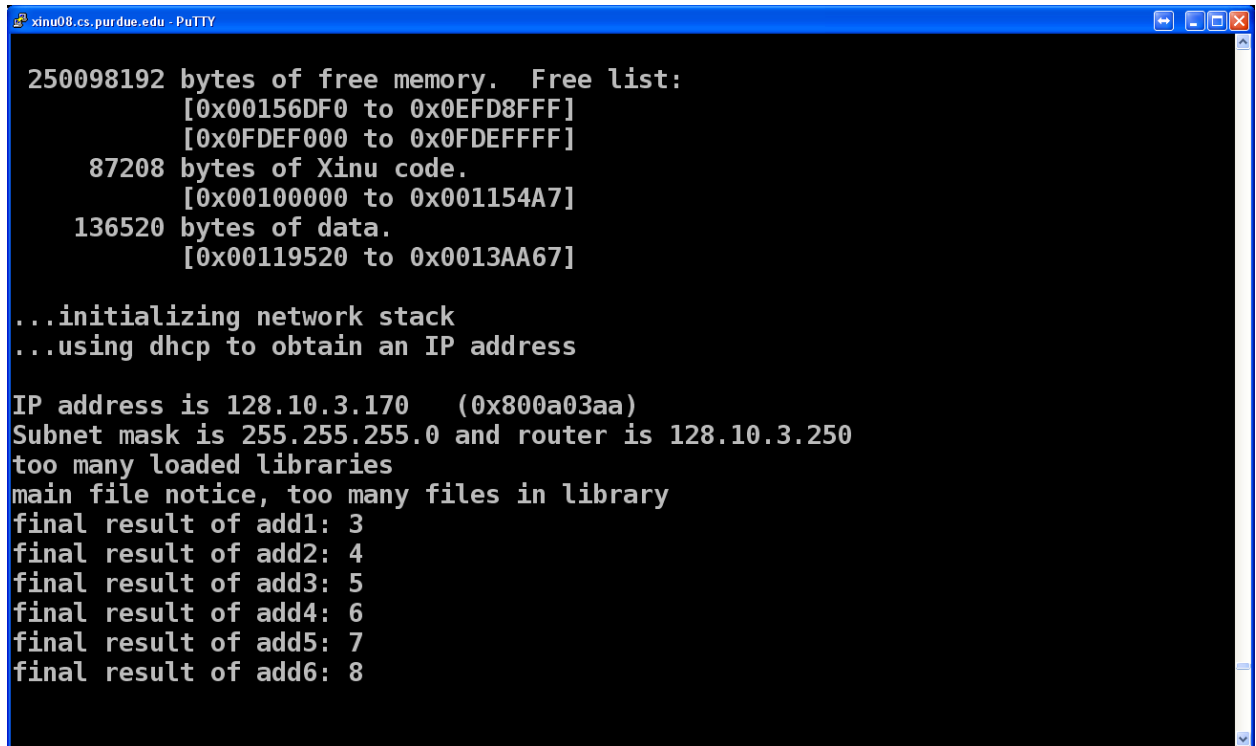
IP address is 128.10.3.161   (0x800a03a1)
Subnet mask is 255.255.255.0 and router is 128.10.3.250
Hello World!
█
```

2. loadlibrary with initial value 2
For load_library, I have initialized a hash table in the xinu system and add the new file functions into the hash table. If I need the functions, I just get from the hash table.
I have several files to load into the library:
myadd.c myadd1.c ... myadd6.c

myadd.c, myadd1.c myadd2.c, myadd3.c implements the functions: add1, add2, ... , add8. Every file has two functions inside.

When I tried to load myadd3.c which is the fourth file, the load_library return syserr and prints the main file notice in main.c.

The function works well while loaded from the libraries.

A screenshot of a PuTTY terminal window titled 'xinu08.cs.purdue.edu - PuTTY'. The terminal displays the following text:

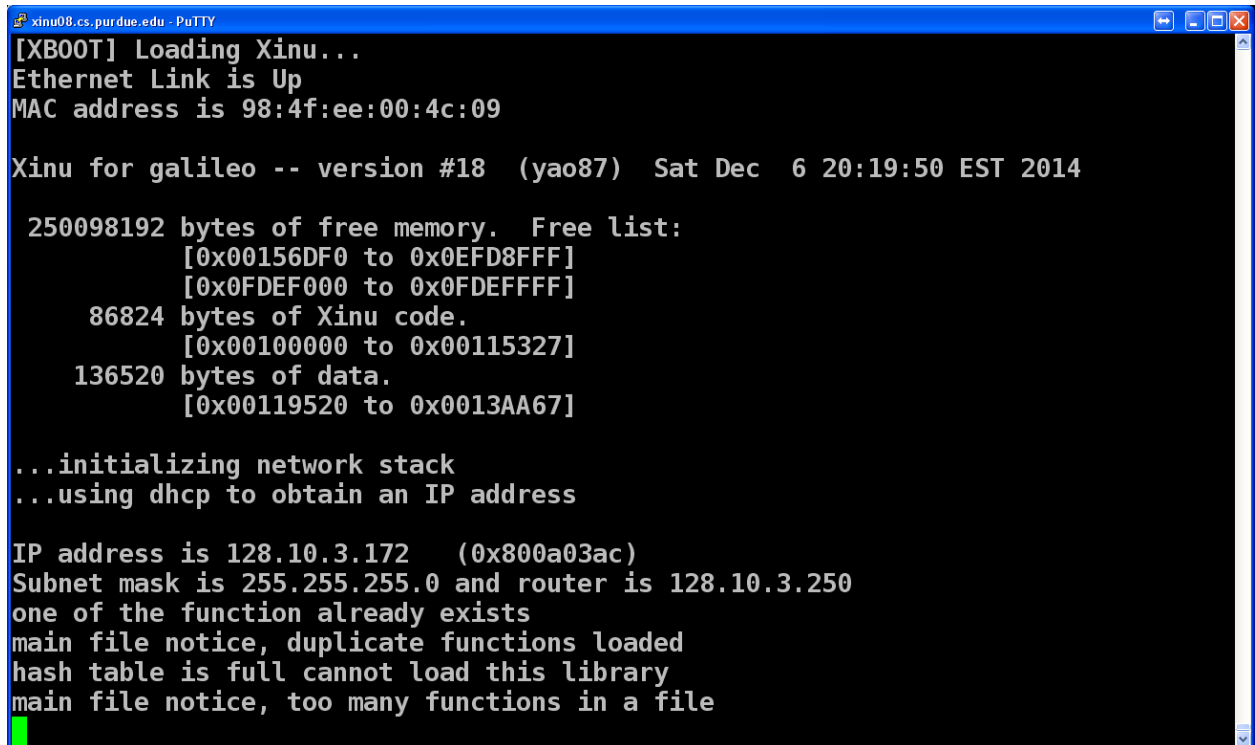
```
250098192 bytes of free memory. Free list:
      [0x00156DF0 to 0x0EFD8FFF]
      [0x0FDEF000 to 0x0FDEFFFF]
  87208 bytes of Xinu code.
      [0x00100000 to 0x001154A7]
136520 bytes of data.
      [0x00119520 to 0x0013AA67]

...initializing network stack
...using dhcp to obtain an IP address

IP address is 128.10.3.170   (0x800a03aa)
Subnet mask is 255.255.255.0 and router is 128.10.3.250
too many loaded libraries
main file notice, too many files in library
final result of add1: 3
final result of add2: 4
final result of add3: 5
final result of add4: 6
final result of add5: 7
final result of add6: 8
```

Myadd5.c file has duplicate add1 which has already been loaded by myadd.c so there would return an error.

Myadd6.c file has so many functions which is more than 10, so there return an error.

A screenshot of a terminal window titled 'xinu08.cs.purdue.edu - PuTTY'. The terminal displays the boot sequence of the Xinu operating system. It starts with '[XB00T] Loading Xinu...', followed by 'Ethernet Link is Up' and 'MAC address is 98:4f:ee:00:4c:09'. The version string 'Xinu for galileo -- version #18 (yao87) Sat Dec 6 20:19:50 EST 2014' is shown. Memory statistics are listed: '250098192 bytes of free memory. Free list:' with two memory ranges, '86824 bytes of Xinu code.' with a range, and '136520 bytes of data.' with a range. The network stack is initialized, and an IP address '128.10.3.172 (0x800a03ac)' is obtained via DHCP. The subnet mask is '255.255.255.0' and the router is '128.10.3.250'. Finally, three error messages are printed: 'one of the function already exists', 'main file notice, duplicate functions loaded', and 'hash table is full cannot load this library'. The last message is followed by 'main file notice, too many functions in a file' and a green cursor.

```
xinu08.cs.purdue.edu - PuTTY
[XB00T] Loading Xinu...
Ethernet Link is Up
MAC address is 98:4f:ee:00:4c:09

Xinu for galileo -- version #18 (yao87) Sat Dec 6 20:19:50 EST 2014

250098192 bytes of free memory. Free list:
    [0x00156DF0 to 0x0EFD8FFF]
    [0x0FDEF000 to 0x0FDEFFFF]
86824 bytes of Xinu code.
    [0x00100000 to 0x00115327]
136520 bytes of data.
    [0x00119520 to 0x0013AA67]

...initializing network stack
...using dhcp to obtain an IP address

IP address is 128.10.3.172 (0x800a03ac)
Subnet mask is 255.255.255.0 and router is 128.10.3.250
one of the function already exists
main file notice, duplicate functions loaded
hash table is full cannot load this library
main file notice, too many functions in a file
█
```

3. Shell implementation

Ls:

Filter out the “..” and “.” when print out, but can still use ls ‘.’ to list the self directory.

```
xinu08.cs.purdue.edu - PuTTY

XINU

-----

Welcome to Xinu!

xsh $ ls
helloworld.c
Makefile
ls
ls.c
semdump.c
testdir/
helloworld
xinu.elf
#helloworld#
myadd
myadd.c
semdump
rfserver
```

Ls directory and ls to a file which returns panic, ls ‘..’ is not working, but ls ‘.’ works well. ls testdir(created directory which has testdir.c) it will return the testdir.c file which is in testdir.

```
xinu08.cs.purdue.edu - PuTTY

xsh $ ls testdir
testdir.c
xsh $ ls myadd
myadd is a file, cannot be read
xsh $ ls .
helloworld.c
Makefile
ls
ls.c
semdump.c
testdir/
helloworld
xinu.elf
#helloworld#
myadd
myadd.c
semdump
rfserver
xsh $ ls ..
No Such Directory or File
```

Semdump:

For the semdump, I have implemented a function in xinu/system which return the semtab, since semtab is not in the xinu.elf file, it’s necessary to implement this file and relocate to find the semaphore table.

```
xinu08.cs.purdue.edu - PuTTY

Welcome to Xinu!

xsh $ semdump
Entry  State    Count  Queue
0      S_USED    0      100
1      S_USED   64      102
2      S_USED   16      104
3      S_USED   -1      106
4      S_USED   64      108
5      S_USED   -1      110
6      S_USED    1      112
7      S_USED    1      114
8      S_USED    1      116
9      S_USED    1      118
10     S_USED    1      120
11     S_USED    1      122
12     S_USED    1      124
13     S_USED    1      126
14     S_USED  128      128
15     S_USED   -1      130
16     S_FREE    0         0
```

The states after 16 are all free.

- The details behind your implementation. As part of this discussion write answers to the following questions:
 - The separation between a Xinu ELF file and the running image leads to a potential problem: if the ELF file is changed (Xinu sources are recompiled) after an image starts to run, symbol table addresses in the ELF file may no longer match the locations of items in the running image. How can you ensure the ELF file read at run time matches the image that is executing?

The xinu.elf file has been loaded into the memory which has already been fixed during running process. So if the elf file has changed, the memory won't change, since there is no read operation again in the code. Try to avoid the elf file change, my way of doing this is trying to read this elf as soon as possible in load program code in order to avoid the change of elf file.

- What is the most difficult aspect of the project? Why?

The most difficult aspect of the project is reading the elf file manuals.

The specifications and details in the manual is not enough for understanding the procedures of implementing the relocation of functions if you never did that before. Thanks for TA's help, I finally figure out what S P A stands for in the relocation equation. This is really interesting project and rfservice gives me a brief understanding of remote file system. There would be sometimes unknown bugs, (maybe due to the file operation bugs), sometimes cannot connect to the server so cannot open the file, but the whole process of doing this is full of fun.