How to Add a System Call in MINIX 3.1.8

By Junjie Li

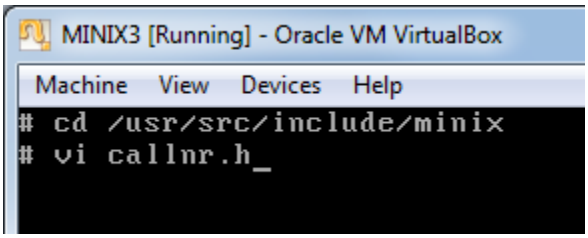
CISE Department, University of Florida

Feb. 13, 2012

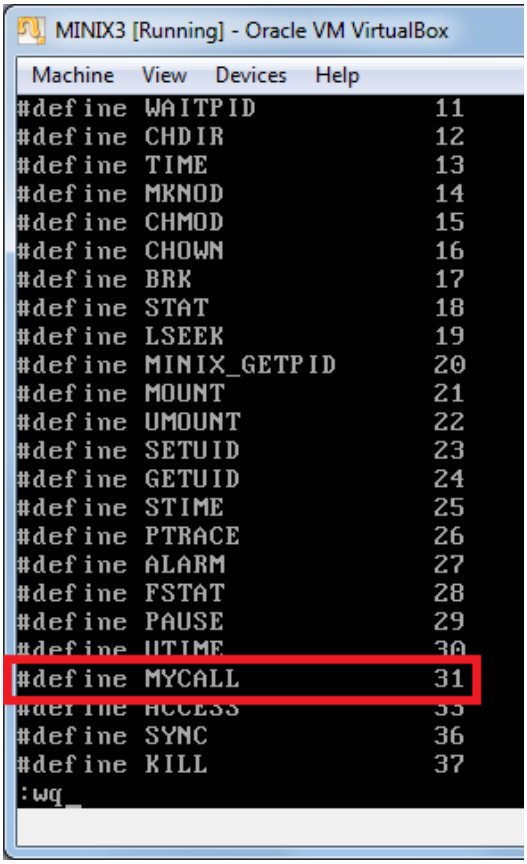
Object: we want to add a simple system call which prints “Hello World! This is my system call!” once called.

Steps:

1. Edit “/usr/src/include/minix/callnr.h” and find an unused slot.



2. Here we use slot 31 and change it to “#define MYCALL 31”.



3. Then edit “/usr/src/servers/pm/table.c”

```
MINIX3 [Running] - Oracle VM VirtualBox
Machine   View   Devices   Help
#define TIME            13
#define MKNOD           14
#define CHMOD           15
#define CHOWN           16
#define BRK             17
#define STAT            18
#define LSEEK           19
#define MINIX_GETPID    20
#define MOUNT           21
#define UMOUNT          22
#define SETUID          23
#define GETUID          24
#define STIME           25
#define PTRACE          26
#define ALARM           27
#define FSTAT           28
#define PAUSE           29
#define UTIME           30
#define MYCALL          31
#define ACCESS          33
#define SYNC            36
#define KILL            37
:wq
# cd /usr/src/servers/pm/
# vi table.c_
```

4. Change slot 31 to “do_mycall”

```
MINIX3 [Running] - Oracle VM VirtualBox
Machine   View   Devices   Help
        do_waitpid,     /* 11 = waitpid */
        no_sys,         /* 12 = chdir   */
        do_time,        /* 13 = time    */
        no_sys,         /* 14 = mknod   */
        no_sys,         /* 15 = chmod   */
        no_sys,         /* 16 = chown   */
        do_brk,         /* 17 = break   */
        no_sys,         /* 18 = stat    */
        no_sys,         /* 19 = lseek   */
        do_get,         /* 20 = getpid  */
        no_sys,         /* 21 = mount   */
        no_sys,         /* 22 = umount  */
        do_set,         /* 23 = setuid  */
        do_get,         /* 24 = getuid  */
        do_stime,       /* 25 = stime   */
        do_trace,       /* 26 = ptrace  */
        do_alarm,       /* 27 = alarm   */
        no_sys,         /* 28 = fstat   */
        do_pause,       /* 29 = pause   */
        no_sys,         /* 30 = utime   */
        do_mycall,              /* 31 = (stty)  */
        no_sys,         /* 32 = (gtty)  */
        no_sys,         /* 33 = access  */
        no_sys,         /* 34 = (nice)  */
:wq_
```

5. Edit “/usr/src/servers/pm/proto.h”

```
MINIX3 [Running] - Oracle VM Virtua
Machine View Devices Help
# pwd
/usr/src/servers/pm
# vi proto.h_
```

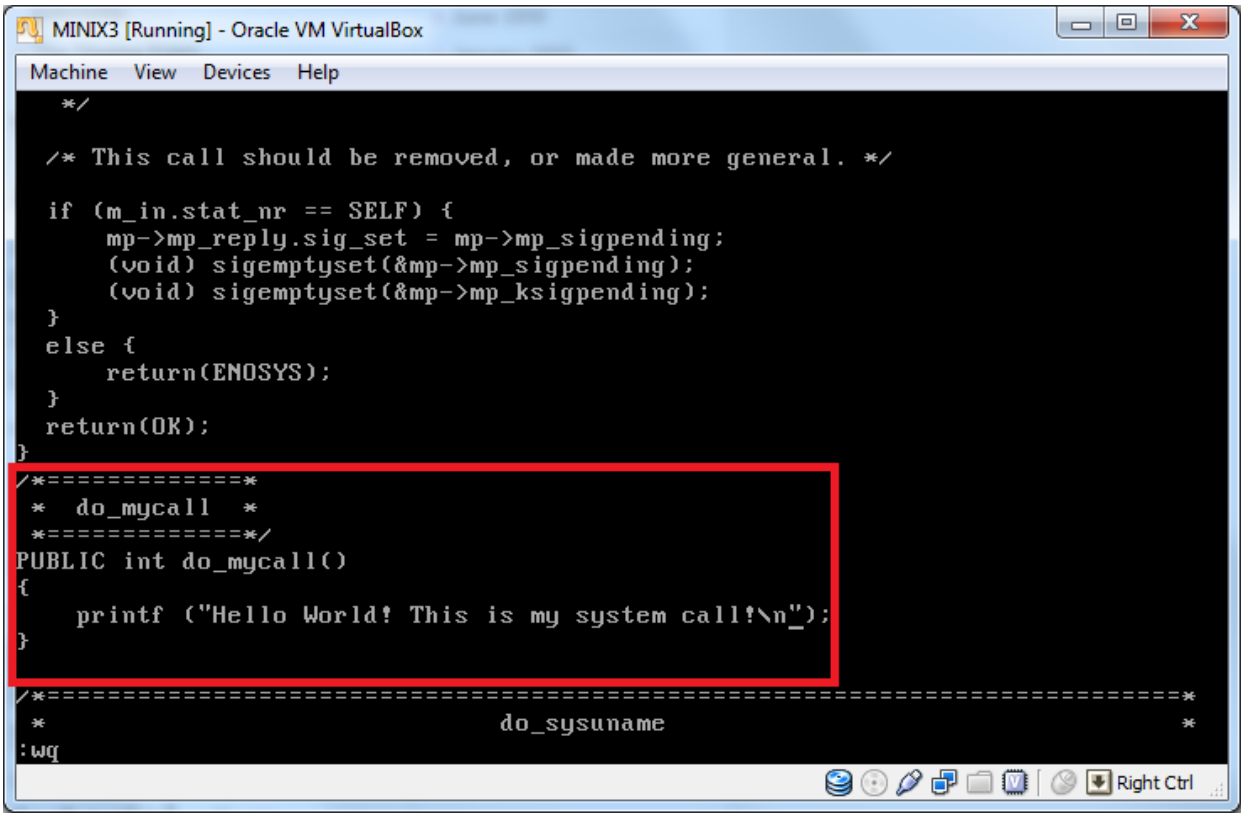
6. Add the prototype in “misc.c” section

```
MINIX3 [Running] - Oracle VM VirtualBox
Machine View Devices Help
/* getset.c */
_PROTOTYPE( int do_get, (void)                                          );
_PROTOTYPE( int do_set, (void)                                          );

/* main.c */
_PROTOTYPE( int main, (void)                                            );
_PROTOTYPE( void setreply, (int proc_nr, int result)                    );

/* mcontext.c */
_PROTOTYPE( int do_getmcontext, (void)                                  );
_PROTOTYPE( int do_setmcontext, (void)                                  );

/* misc.c */
_PROTOTYPE( int do_reboot, (void)                                       );
_PROTOTYPE( int do_procstat, (void)                                     );
_PROTOTYPE( int do_sysuname, (void)                                     );
_PROTOTYPE( int do_getsysinfo, (void)                                   );
_PROTOTYPE( int do_getsysinfo_up, (void)
_PROTOTYPE( int do_getprocnr, (void)                                    );
_PROTOTYPE( int do_getepinfo, (void)                                    );
_PROTOTYPE( int do_svrctl, (void)                                       );
_PROTOTYPE( int do_getsetpriority, (void)                               );
_PROTOTYPE( int do_mycall, (void)                                       );

:wq_
Right Ctrl
```

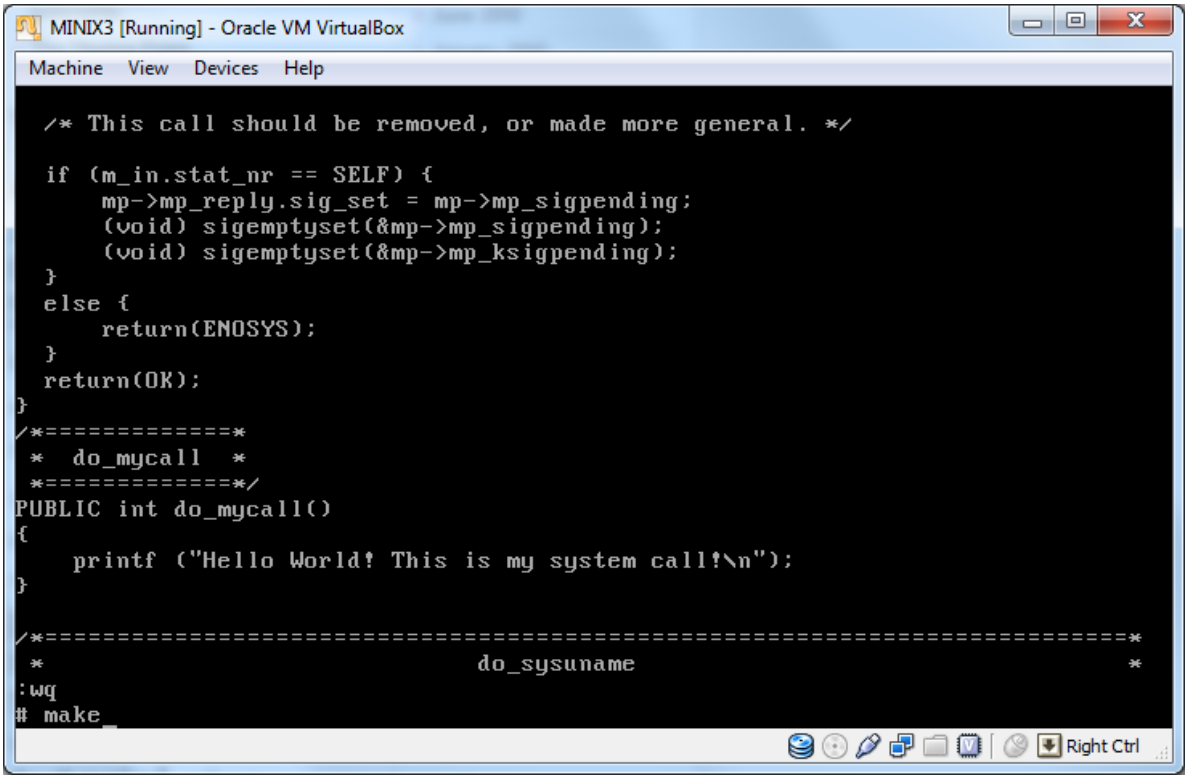
7. Edit “/usr/src/servers/pm/misc.c”

```
MINIX3 [Running] - Oracle VM VirtualBox
Machine View Devices Help

/* main.c */
_PROTOTYPE( int main, (void)                                            );
_PROTOTYPE( void setreply, (int proc_nr, int result)                    );

/* mcontext.c */
_PROTOTYPE( int do_getmcontext, (void)                                  );
_PROTOTYPE( int do_setmcontext, (void)                                  );

/* misc.c */
_PROTOTYPE( int do_reboot, (void)                                       );
_PROTOTYPE( int do_procstat, (void)                                     );
_PROTOTYPE( int do_sysuname, (void)                                     );
_PROTOTYPE( int do_getsysinfo, (void)                                   );
_PROTOTYPE( int do_getsysinfo_up, (void)
_PROTOTYPE( int do_getprocnr, (void)                                    );
_PROTOTYPE( int do_getepinfo, (void)                                    );
_PROTOTYPE( int do_svrctl, (void)                                       );
_PROTOTYPE( int do_getsetpriority, (void)                               );
_PROTOTYPE( int do_mycall, (void)                                       );

:wq
# pwd
/usr/src/servers/pm
# vi misc.c_
Right Ctrl
```

8. Add the definition of the call in “misc.c”



9. Execute “make” to check if the definition compiles.



10. Create a file named “mycalllib.h” under “/usr/include”.

```
MINIX3 [Running] - Oracle VM VirtualBox
Machine   View   Devices   Help
     create  pm/signal.d
     create  pm/table.d
     create  pm/time.d
     create  pm/trace.d
     create  pm/utility.d
     create  pm/.depend
    compile  pm/main.o
    compile  pm/forkexit.o
    compile  pm/break.o
    compile  pm/exec.o
    compile  pm/time.o
    compile  pm/alarm.o
    compile  pm/signal.o
    compile  pm/utility.o
    compile  pm/table.o
    compile  pm/trace.o
    compile  pm/getset.o
    compile  pm/misc.o
    compile  pm/profile.o
    compile  pm/dma.o
    compile  pm/mcontext.o
    compile  pm/schedule.o
       link  pm/pm
# cd /usr/include/
# vi mycalllib.h
```

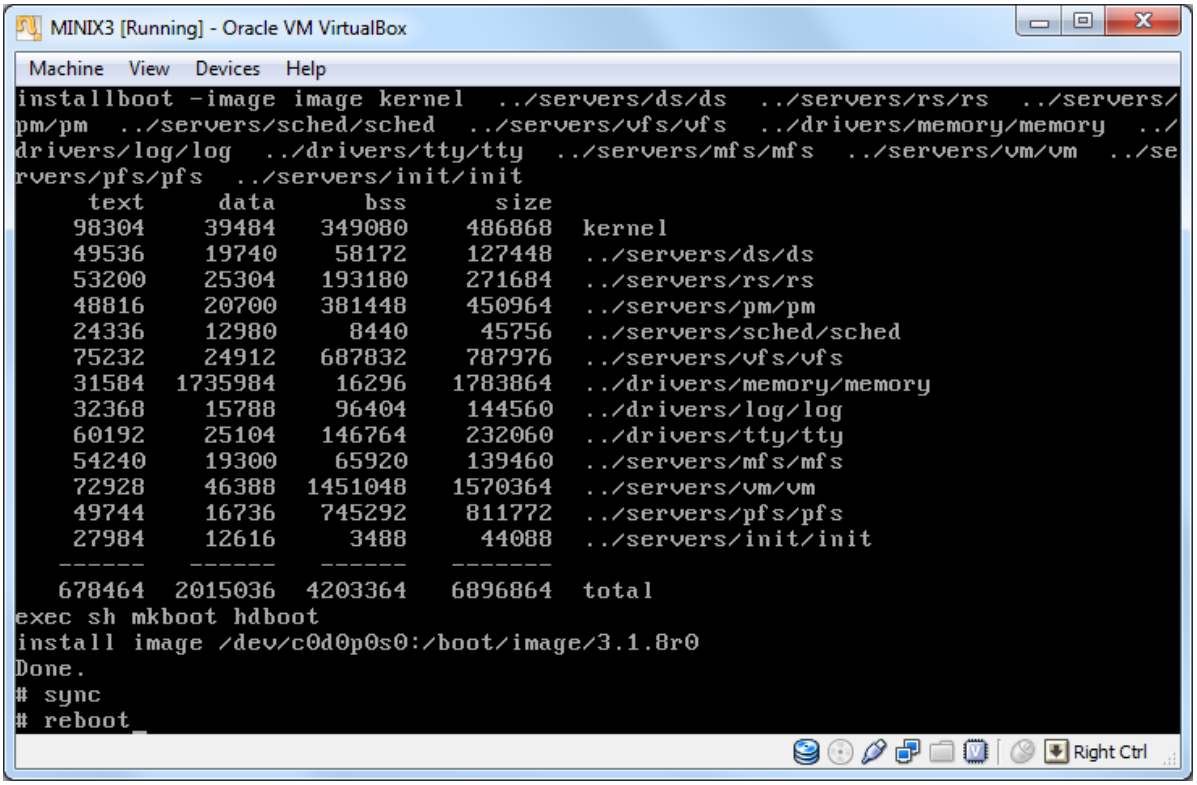
11. The content of the file “mycalllib.h”

```
MINIX3 [Running] - Oracle VM VirtualBox
Machine   View   Devices   Help
#include <lib.h>
#include <unistd.h>

PUBLIC int mycall ()
{
    message m;
    return ( _syscall(PM_PROC_NR, MYCALL, &m) );
}
~
```

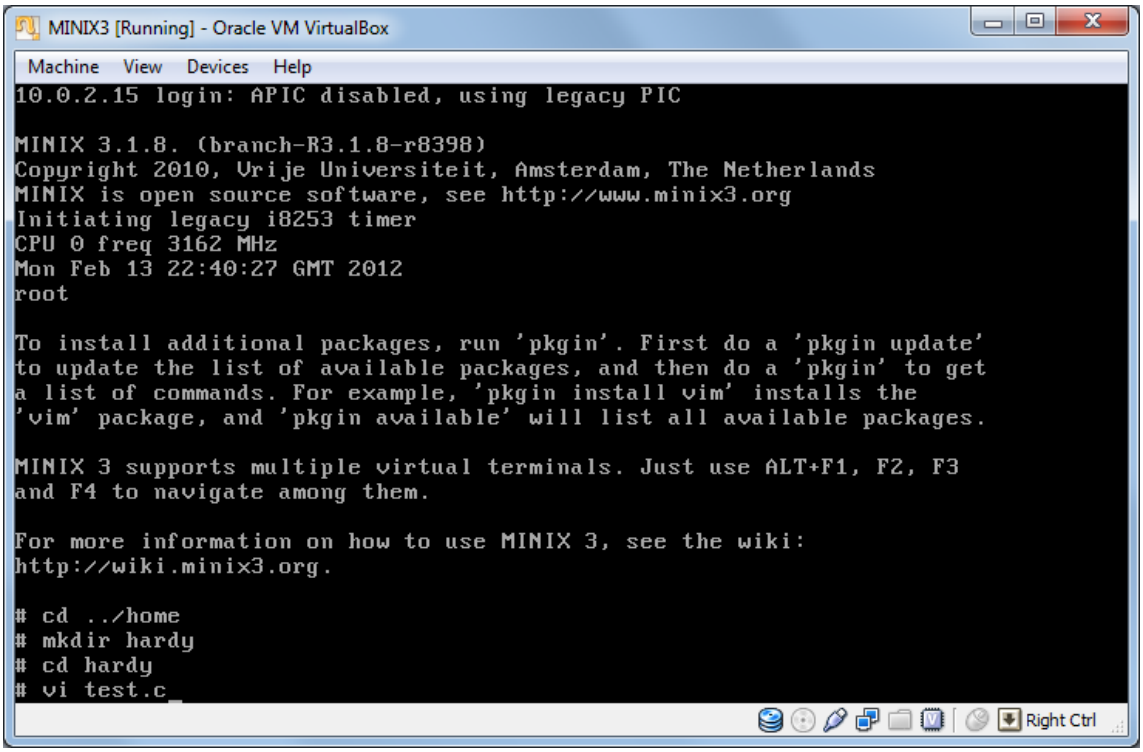
12. Compile the new system.

```
# cd /usr/src/tools
# make hdboot
```

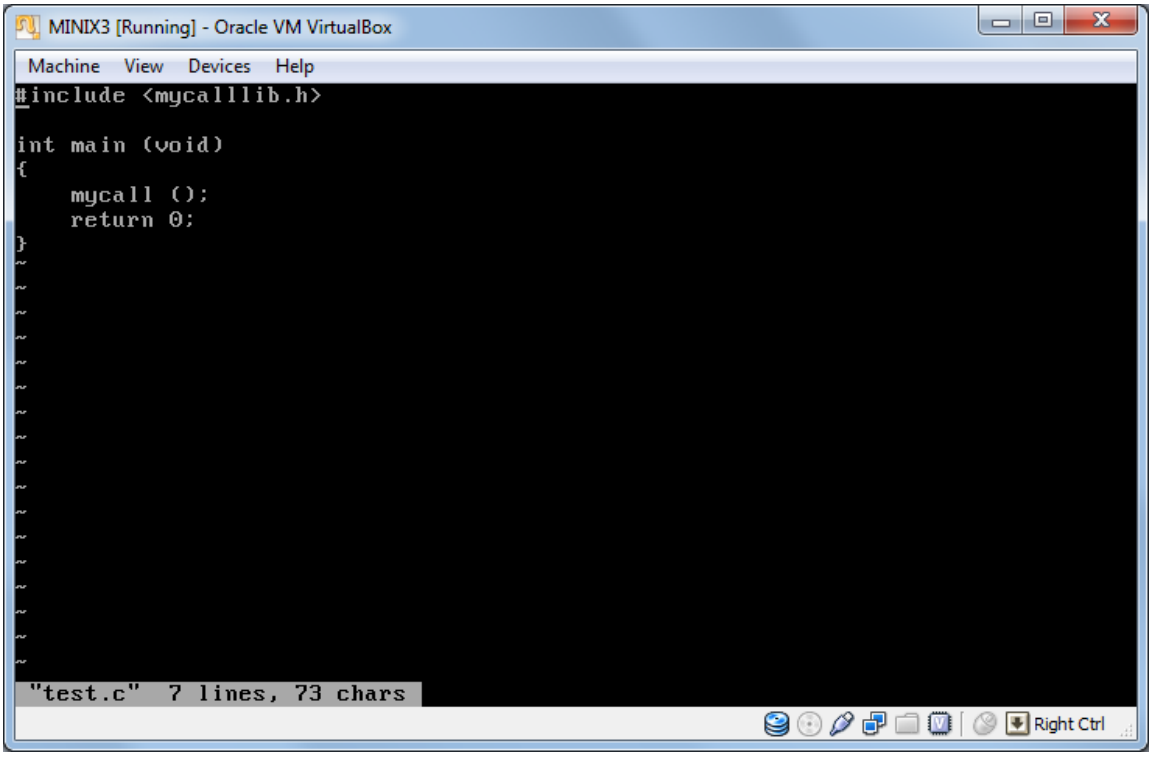
13. “sync” and “reboot”



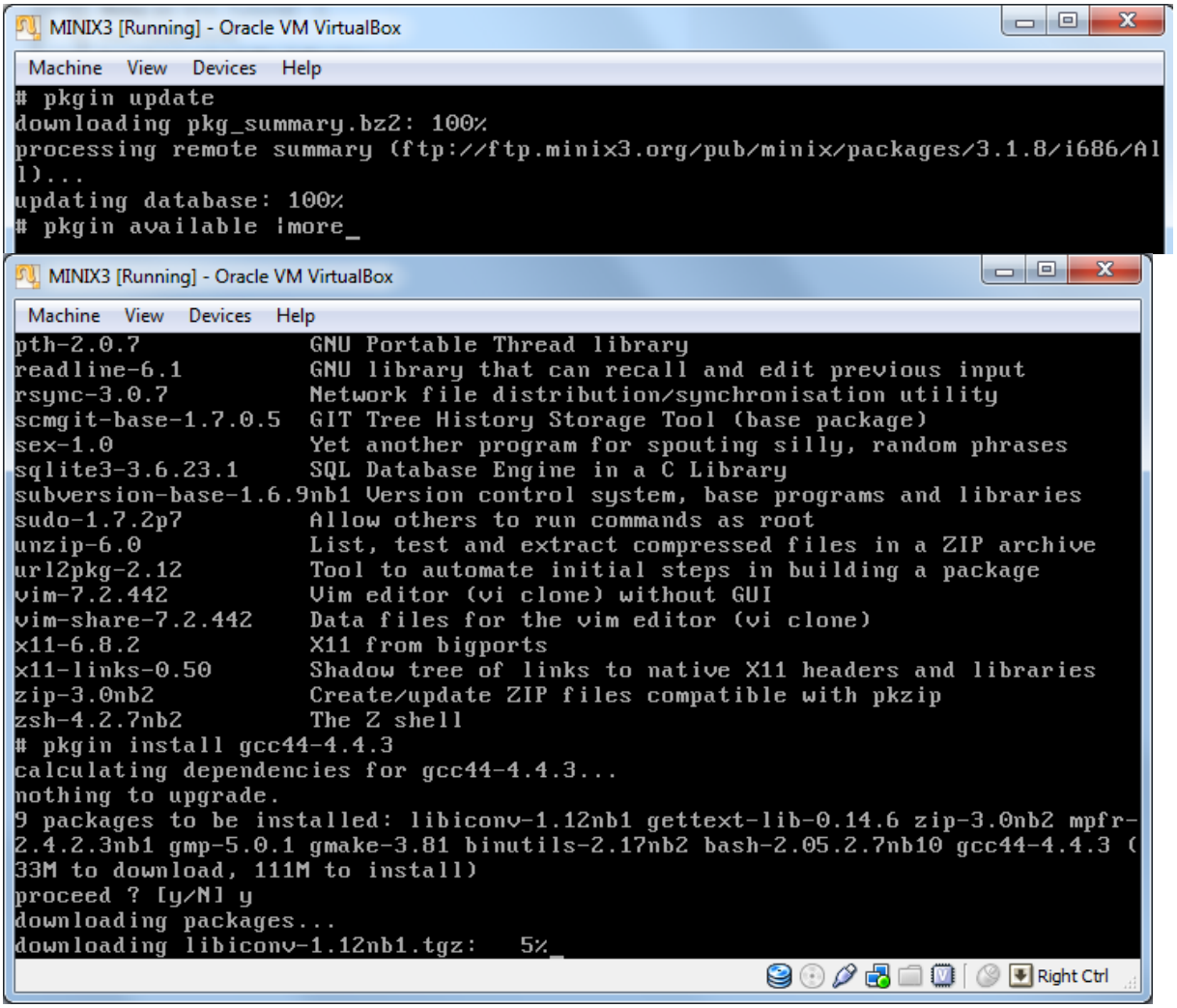
14. Compose a test file.



15. The test file looks like this:



16. Install gcc by “pkgin update” and “pkgin install gcc44-4.4.3”.



17. Compile and run. You should all be set!