Installation Guide for OS/161

Step 1: You will need the following source files. Please make sure that you already downloaded them.

- os161.tar.gz
- os161-binutils.tar.gz
- os161-gcc.tar.gz
- os161-gdb.tar.gz
- sys161.tar.gz

Step 2: Build and Install the Binary Utilities (Binutils)

Unpack the binutils archive by executing the following command **inside your Download folder**:

```
tar -xzf os161-binutils.tar.gz
```

```
Next, In binutils-2.17+os161-2.0.1/configure
Change
| egrep 'texinfo[^0-9]*([1-3][0-9]|4\.[4-9]|[5-9])' >/dev/null 2>&1; then

to

| egrep 'texinfo[^0-9]*([1-3][0-9]|4\.([4-9]|[1-9][0-9])|[5-9])' >/dev/null 2>&1; then
```

Please note the change is made only in the part that is underlined.

Next, move into the newly-created directory:

```
cd binutils-2.17+os161-2.0.1
```

Configure binutils by executing the following command **(Please note that the following needs to be typed in a single line)**

```
./configure --nfp --disable-werror --target=mips-harvard-os161 --prefix=$HOME/sys161/tools
```

Run the following command:
```
find . -name '*.info' | xargs touch
```

Make binutils by executing the following command:
```
make
```

Finally, once `make` has succeeded, install the binutils into their final location:
```
make install
```
This will create the directory `$HOME/sys161/tools/` and populate it.

Step 3: Adjust Your Shell's Command Path

First, make the directory in which your shell will eventually find the toolchain binaries:
```
mkdir $HOME/sys161/bin
```
Next, add two directories (`$HOME/sys161/bin` and `$HOME/sys161/tools/bin`) to your shell's search path.

```
export PATH=$HOME/sys161/bin:$HOME/sys161/tools/bin:$PATH
```

Note that setting these variables only works in the shell or window where you execute the above commands. You will need to re-execute these command if you reboot Ubuntu or open a new shell to ensure that the proper path is set and used for future logins and for other newly created shells.

Note that you may need to log out and log back in again so that this `PATH` change will take effect. You can check the current setting of the `PATH` environment variable using the command

```
printenv PATH
```

Step 4: Install the GCC MIPS Cross-Compiler

Unpack the gcc archive by executing the following command **inside your Download folder**:

```
tar -xzf os161-gcc.tar.gz
```

```
In gcc-4.1.2+os161-2.0/configure,
Change
 | egrep 'texinfo[^0-9]*([1-3][0-9]|4\.[2-9]|[5-9])' >/dev/null 2>&1; then
```

```
to
```

```
| egrep 'texinfo[^0-9]*([1-3][0-9]|4\.([2-9]|[1-9][0-9])|[5-9])' >/dev/null 2>&1; then
```

Please note the change is made only in the part that is underlined.

Move into the newly-created directory:

```
cd gcc-4.1.2+os161-2.0
```

Configure gcc by executing the following command (**Please note that the following needs to be typed ins a single line**)

Please note that "mips-harvard-os161" in the following command needs to be typed as a single word.

```
./configure -nfp --disable-shared --disable-threads --disable-libmudflap --disable-libssp --target=mips-
harvard-os161 --prefix=$HOME/sys161/tools
```

Make it and install it:

```
make
make install
```

Step 5: Install GDB

Unpack the gdb archive by executing the following command **inside your Download folder**:

```
tar -xzf os161-gdb.tar.gz
```

```
In gdb-6.6+os161-2.0/configure
Change
| egrep 'texinfo[^0-9]*([1-3][0-9]|4\.[4-9]|[5-9])' >/dev/null 2>&1; then

to
| egrep 'texinfo[^0-9]*([1-3][0-9]|4\.([4-9]|[1-9][0-9])|[5-9])' >/dev/null 2>&1; then
```

Please note the change is made only in the part that is underlined.

Move into the newly-created directory:

```
cd gdb-6.6+os161-2.0
```

Configure gcc

./configure --target=mips-harvard-os161 --prefix=$HOME/sys161/tools --disable-werror

```
make
make install
```

Step 6: Build and Install the sys161 Simulator

Unpack the sys161 archive by executing the following command **inside your Download folder**:

```
tar -xzf sys161.tar.gz
```

Move into the newly-created directory:

```
cd sys161-1.99.05
```

Next, configure sys161:

```
./configure --prefix=$HOME/sys161 mipseb
```

Build sys161 and install it:

```
make
make install
```

Step 7: Set Up Links for Toolchain Binaries

```
cd $HOME/sys161/tools/bin
```

The following needs to be typed as a single line. Please note that "-f4-" needs to be typed as a single word. Also, note that ' and ` are different characters in the following command.

```
sh -c 'for i in mips-*; do ln -s ../tools/bin/$i ~/sys161/bin/cs4300-`echo $i | cut -d- -
f4-`; done'
```

```
cd $HOME/sys161
```

```
ln -s share/examples/sys161/sys161.conf.sample sys161.conf
```

When you are finished with these steps, a listing of the directory `$HOME/sys161/bin` should look similar to this:

```
cs4300-addr2line@  cs4300-gcc-4.1.2@  cs4300-nm@        cs4300-size@       stat161-1.99.05*
cs4300-ar@         cs4300-gccbug@     cs4300-objcopy@  cs4300-strings@    sys161@
cs4300-as@         cs4300-gcov@       cs4300-objdump@  cs4300-strip@      sys161-1.99.05*
cs4300-c++filt@    cs4300-gdb@        cs4300-ranlib@   hub161@           trace161@
cs4300-cpp@        cs4300-gdbtui@     cs4300-readelf@  hub161-1.99.05*  trace161-1.99.05*
cs4300-gcc@        cs4300-ld@         cs4300-run@      stat161@
```

These are all of the tools you will need to work with sys161.

Step 8: Install OS/161

First, create a directory to hold the OS/161 source code, your compiled OS/161 kernels, and related test programs.

```
cd $HOME
mkdir cs4300-os161
```

Next, move the OS/161 archive into your new directory and unpack it:

```
mv os161.tar.gz cs4300-os161
cd cs4300-os161
tar -xzf os161.tar.gz
```

This will create a directory called `os161-1.11` (under `cs4300-os161`) containing the OS/161 source code.

Configure and Build OS/161

Step 1: Configure and Build OS/161

The next step is to configure OS/161 and compile the kernel. From the `cs4300-os161` directory, do the following:

```
cd os161-1.11


./configure --ostree=$HOME/cs4300-os161/root --toolprefix=cs4300-


cd kern/conf


./config ASST0


cd ../compile/ASST0


make depend
make
make install
```

The string *ASST0* in the commands above indicates that you are working on assignment 0. For Assignment *X*, replace *ASST0* with *ASSTX* in the commands above. The `make install` command will create a directory called `$HOME/cs4300-os161/root` (`$HOME` refers to your home directory), into which it will place the compiled kernel in a file called `kernel-`*`ASST0`*. It will also create a symbolic link call `kernel` referring to `kernel-`*`ASST0`*. Check the `$HOME/cs4300-os161/root` directory to make sure that your kernel is in place.

Next, build the OS/161 user level utilities and test programs:

```
cd $HOME/cs4300-os161/os161-1.11
make
```

Step 2: Try Running OS/161

You should now be able to use the SYS/161 simulator to run the OS/161 kernel that you built and installed. The SYS/161 simulator requires a configuration file in order to run. To obtain one, do this:

```
cd $HOME/cs4300-os161/root
cp $HOME/sys161/sys161.conf .
```
(Please note the "." at the end of the command which is required)

Now run your compiled kernel on the simulator. Assuming that you're still in the `$HOME/cs4300-os161/root` directory, do this:

```
sys161 kernel-ASST0
```

You should see some output that looks something like this:

```
sys161: System/161 release 1.99.05, compiled Apr 28 2011 21:49:59

OS/161 base system version 1.11
Copyright (c) 2000, 2001, 2002, 2003
   President and Fellows of Harvard College.  All rights reserved.

Put-your-group-name-here's system version 0 (ASST0 #1)

Cpu is MIPS r2000/r3000
336k physical memory available
Device probe...
lamebus0 (system main bus)
emu0 at lamebus0
ltrace0 at lamebus0
ltimer0 at lamebus0
hardclock on ltimer0 (100 hz)
beep0 at ltimer0
rtclock0 at ltimer0
```

```
lrandom0 at lamebus0
random0 at lrandom0
lhd0 at lamebus0
lhd1 at lamebus0
lser0 at lamebus0
con0 at lser0
pseudorand0 (virtual)
OS/161 kernel [? for menu]:
```

The last line is a command prompt from the OS/161 kernel. For now, just enter the command `q` to shut down the simulation and return to your shell.