Simon Fraser University
School of Engineering Science
ENSC351: Real Time and Embedded Systems
Fall 2011

Makefiles for Linux

Makefiles are a common construct for generating executables in a Unix-type environment such as Linux. They provide the user with greater control over the compilation process. Below is an example of a simple Makefile for a "Hello World" application.

```
all: hello_world.exe

hello_world.exe: hello_world.o
	qcc –V gcc_ntox86 –g –o hello_world hello_world.o

hello_world.o : hello_world.c
	qcc –V gcc_ntox86 –c –g hello_world.c
clean:
	rm hello_world.o hello_world.exe
```

To run the commands in a Makefile, the user simply types `make <option>` at the command prompt, where the `option` is one of headings (in this example `all`, `hello_world.exe`, `hello_world.o`, and `clean`). For example, `make clean` will run the `rm` command deleting the hello_world.o and hello_world.exe files. By default, if the user types only `make`, the command `make all` is run.

The gcc command is the standard GNU Linux C compiler. In this sample makefile, I've used the following options:

the –`V` flag allows the user to specify the compiler, version, and target for the compiler
the –`c` flag causes the compiler to only compile only (generate an object file and not an executable)
the –`g` flag ensures that the compiler includes debug information in the executable
the –`o` flag allows the user to specify the output filename

For a summary of all the gcc compiler flags, you can simply google gcc.

****Important Note***: The formatting for Makefiles is finicky. All indentations are done with ***tabs*** and not ***spaces***. If you use spaces, and not tabs, your Makefile will ***not*** work.