Makefiles

Michael Brodskiy
Professor: S. Shazli

April 10, 2023

- Header files consist of definitions of functions and declarations of variables
- `#ifndef` is used to define something in the code if it is not already defined (usually used for constants)
- `#define` is used to just define something
- When compiling multiple files, the following rules apply:
  - Compiled:
    - .c or .cpp
  - Not Compiled:
    - .h or .hpp
- `make` is a tool that is designed to allow programmers to efficiently compile large complex programs with many components easily
- The `make` utility allows us to only compile those that have changed and the modules that depend upon them
- If any of the associated files have been modified, then it recompiles
- If the file `<target>` does not exist, or the dependency files are younger then execute `<buildCommand>`
- To run a makefile and compile a program, we run the command `make`
- The `make` command will build the first target which is the only target in this example
- If you want to build a specific target if there are multiple targets, you can specify the target with the `make` command
- A `clean` target in a makefile removes or cleans the system of specified files
  - Executed using `make clean`
- Variables/symbols may be used to make the makefile easier to construct
  - For example, use something like `PROJECT = homework2` as a variable, and then `$(PROJECT)` to refer to it
  - "Phony" targets do not create files, will always be executed when called
  - Flags and compiler may be specified as well
- A makefile will be necessary for the final project
- The `LEDControl` class from the previous lab will be modified to consist of multiple files, which create an object `DE1Socfpga` with a header file

- This will then be applied to seven segment displays
- The displays will be controlled by objects with header files as well
- Bit manipulation will be necessary to control the displays