Changes to page "Practical: Makefile and wiringPi C++ Programming Requirements"

James Fischer

All Sections

During the practical portion of the final exam in ECE 3220 you must create a Makefile that you will then use to build your software on the Raspberry Pi.

When I described this makefile during the ECE 3220 lecture hour I made a false assumption, and consequently I did not mention one additional/important makefile rule which I have subsequently added to the page **Practical: Makefile and wiringPi C++ Programming Requirements**.

Specifically, the makefile I described in class assumes your project does not have any header files (%.h). For the practical, your software project WILL have a header file named `rpi3b_accessory.h`. Therefore, your makefile must be modified slightly to ensure the project's object code files (%.o) are rebuilt whenever (1) a corresponding %.cc file is edited/updated, or (2) the header file `rpi3b_accessory.h` is edited/updated. Case (1) is already covered by the makefile I described during the lecture.

To cover case (2), add a new variable named HEADERS whose value is the set of header file names that reside within the current working directory. In your case this will be the one and only header file name, `rpi3b_accessory.h`:

```
HEADERS := $(wildcard *.h)
```

Next, add a new rule *AFTER* the build rule for your PROGRAM that defines a dependency between the project's object code files OBJS and the project's header files HEADERS--i.e., if any header file is edited/updated, the makefile must ensure the project's object code files OBJS are updated by recompiling them (see the yellow-highlighted section below), and the updated OBJS files then trigger a rebuild of the PROGRAM file from the updated OBJS files:

```
# Build rule
$(PROGRAM) : $(OBJS)
        $(LINK.o) $^ $(LDLIBS) -o $@

# Inform make that the project's object code files %.o must
# be rebuilt if any header file is edited/updated.
$(OBJS) : $(HEADERS)
```