Policies Regarding Project Hardware

1.0 Introduction:

ECE477 is a senior design course that is designed to motivate and challenge students, as well as simulate the electronic design process in a professional setting. In order to provide sufficient challenge and maintain professionalism, certain hardware is and is not allowed to be used in ECE477 senior design. This course policy document serves to inform students about acceptable and unacceptable final hardware for use in their ECE477 senior design projects. Final hardware and prototyping hardware are defined in section 1.1, below.

1.1 Prototyping Hardware and Final Hardware

For the purposes of this course, prototyping hardware shall refer to any hardware used to test devices, software, concepts, or functions in a preliminary state. No restrictions are placed on hardware used to prototype for ECE477; students may use whatever development hardware and tools they desire.

Final hardware refers to the hardware that is used in the final senior design project, and refers to any hardware used to satisfy preliminary or final PSSCs for the course. Final hardware is subject to the rules described in this document. PSSCs can only be satisfied by projects utilizing valid final hardware; projects utilizing prototyping hardware for a particular function will be unable to satisfy PSSCs and will be insufficient for passing course outcomes.

2.0 Rules Concerning Final Hardware

The rules concerning valid final hardware for ECE477 are listed below:

1. The ECE477 staff shall have the sole determination of what is and what is not valid final hardware for an ECE477 project.
2. In order to pass preliminary or final project-specific success criteria (PSSCs), a student team must use final hardware to realize the functions being tested in their PSSC (in the case of preliminary PSSCs) or their complete system (in the case of final PSSCs). Protyping hardware will not be acceptable for realizing a given PSSC and will be insufficient to satisfy a given course outcome.
3. Student teams are required to adhere to final hardware rules concerning development boards and prototyping microcontrollers. These rules are elaborated on in section 2.1.
4. Student teams are required to adhere to final hardware rules concerning breakout boards and electronic shortcuts. These rules are elaborated on in section 2.2.
5. Student teams are required to adhere to final hardware rules concerning development IDEs. These rules are elaborated on in section 2.3.
6. Exceptions and modifications to these rules can be made at the sole discretion of the ECE477 course staff. Students wishing to make such exceptions to policy must have written permission from the ECE477 course staff to do so.

2.1 Course Policy Concerning Development Boards and Prototyping Microcontrollers

At the time of this writing, there are a number of development systems and prototyping

devices created and sold by third parties, often catering to the hobbyist market. Popular contemporary examples of such devices include the Basic Stamp, the Arduino and its many variants, the Chipkit, the Flora development system sold by Adafruit, and many others. A general common feature of these devices is that they feature a printed circuit board, a microcontroller, microprocessor, or FPGA, passive components and/or ICs, and optionally a programming interface, such as JTAG, USB, or a breakout ICSP programming header. These prototyping systems may additionally include add-on daughterboard or mezzanine cards to expand their functionality (Arduino shields are a popular example of this, though other examples exist).

As these systems feature significant overhead to a microcontroller and/or printed circuit board performing the same functionality, these development systems may be appropriate for the purposes of prototyping but are inappropriate for use in final products on the market. In addition, students are expected to design, create, and assemble their own printed circuit boards, featuring a microcontroller in this class (a function which development boards fulfill). **As such, development and prototyping systems are banned for the purposes of final project hardware.** There are exceptions to this rule, and these are described below.

There are certain situations where the use of development and prototyping systems may be acceptable for final hardware. One such exception is made for motherboards and systems running embedded operating systems. Examples of such systems include the Raspberry Pi, BeagleBoard/BeagleBone, Pandaboard, Intel Atom Motherboards, and others. Another possible exception is made for systems that perform operations that are deemed to be computationally complex. An example of a computationally complex system that is generally allowed to be used as final hardware is an inertial measurement unit (IMU) or flight controller used in UAVs and other projects. Yet another exception is made for boards in which the functionality necessary for the student project is only available in a device with very difficult soldering or layout requirements. The primary area where this exception comes into play is in projects involving FPGAs, which often come in BGA or other packages that are not hand-solderable, and often require multilayer boards. If a student team believes their hardware qualifies for any of these exceptions, they must contact course staff to receive approval.

2.2 Course Policy Concerning Breakout Boards and Electronic Shortcuts

At the time of this writing, a number of third-party vendors exist that provide breakout boards and electronic shortcuts, primarily to hobbyists. Examples of such vendors include Sparkfun, Adafruit, and Seeed Studio, among many others. These devices generally consist of a relatively small and simple printed circuit board, containing a small number of integrated circuits and passive components, and often feature headers or plated holes to make the parts easy to solder or otherwise integrate into a system.

As these systems constitute significant overhead to the basic parts laid out on a custom circuit board, these electronic shortcuts may be appropriate for purposes of prototyping but are inappropriate for use in final products on the market. Additionally, students are expected to design and build custom circuit boards which are motivating and challenging to them. **As such, breakout boards and electronic shortcuts are banned for the purposes of final project hardware.** There are exceptions to this rule, and these are described below.

Under certain situations, the use of raw components may be unreasonable or unfeasible in the time constraints of ECE477. In these instances, the use of breakout boards and electronic shortcuts may be acceptable. One such instance is when the functionality being performed by the breakout board or shortcut is deemed electromagnetically or procedurally complex. Common

examples of where this shortcut applies include breakout boards that handle the functionality of a wireless interface (Bluetooth, Zigbee, RF, Wifi, GPS receivers, and others), a complex power regulator (some switching power supplies), or USB-to-serial interface ICs used in conjunction with certain microcontrollers. In these instances, the use of a drop-in module is generally allowed and even recommended. Another situation where breakout boards and electronic shortcuts are allowed are situations where the functionality necessary for the student project is not otherwise readily available in a component or components that are hand-solderable. Examples of this exception include certain ICs that are only available in ball grid array packages, and many times certain sensors such as accelerometers magnetometers, and pressure sensors are only available in packages which are extremely difficult or impossible to solder by hand. If a student team believes their hardware qualifies for an exception, they must contact course staff to receive approval.

2.3 Course Policy Concerning Development IDEs

In addition to the development systems featured in section 2.1, a number of third-party IDEs have been developed that accompany these systems. These IDEs often free the developer from having to write low-level code in languages like C and instead allow the creation of code in more abstracted languages such as Python, Perl, custom third-party languages, or even graphical GUIs. One of the best contemporary examples of such an IDE is the Arduino IDE, which can be used to develop “sketches” on certain Atmel chips in place of the lower-level, more general purpose AVRStudio IDE.

As the languages used by these tools are often highly abstracted, the code produced by development IDEs of this nature often contains excessive overhead and performs poorly when compared to comparable code written in a lower level language and compiled. Additionally, IDEs of this nature are considered low-effort and don't adequately simulate the software tools and tool-related challenges electrical engineering graduates are expected to use and deal with in the vast majority of professional settings. **As such, the use of development IDEs of this nature are banned for the purposes of final senior design projects.** If a student team believes their software development tools qualify for an exception to this rule, they must contact course staff to receive approval.