Electronics II
exercise 2

Deadline: Wed 7.2.2024 16:00

Buzzers, interrupts and floats

Interrupts are time or input based functions, which can be used to do execute snippets of code almost instantaneously when a specific event happens.

Input interrupts keep watch on a specific pin for a specific event. When the event happens, the function related to the interrupt runs, regardless of which point the main code (`loop()` and `setup()`) is. After the interrupt is done, the main code will continue running from where it left off as if nothing happened, unless the interrupt altered a variable the code is using.

Using the `delay()` function is an easy way to delay the execution of code. However, using a delay stops all code from executing, so the microcontroller will sit idle. Using a timer interrupt frees the microcontroller to execute other code while waiting for a timer to expire. When the timer expires, the related interrupt function is completed immediately, after which the main code is continued.

Interrupts function outside the main code with elevated priority, so it's best to keep the interrupt code itself very simple by, e.g., setting a value and immediately exiting. Other interrupts, `millis()` and `delay()` (which use interrupts) don't work while an interrupt is executing, so they may run later than expected, return wrong values or not run at all. Another thing of note is that interrupt functions can't return values, so they must alter an existing variable.

Further reading: A more detailed description of how the processor actually handles interrupts (keskeytys) and the privileged execution of code (etuoikeutettu suoritustila) can be found in Tietokoneen toiminnan peruskurssi 2.2 Käskyjen nouto- ja suoritussykli and Tietokoneen toiminnan jatkokurssi 5.2 Tilarekisteri SR.

Task 1 – Tilt alarm (1p)

Use an active buzzer to make a one second beep when a tilt switch is tilted. Realize this by attaching the switch to an interrupt. Make the code such that the interrupt is detached after a set number of beeps (e.g., 3).

More details in the Arduino language reference: attachInterrupt(), detachInterrupt().

NB: The buzzer is a bit quiet when powered with the ESP32's 3.3V GPIO pins.

Task 2 – RGB LED (1p)

Make an RGB LED cycle through red, green, blue, yellow, magenta, and cyan. Use an interrupt attached to a timer instead of the `delay()` function.

Consider operating the LED with port manipulation and Boolean operators. Port manipulation is not required for this exercise but will be useful later (Exercise 4, task 2).

Arduino Timer Interrupts (deepbluembedded.com), for ESP32 (deepbluembedded.com)

Task 3 – Music pad (1+1p)

a) Use the passive buzzer to play Nokia tune or other monophonic melody of your choice. RTTTL files can be used, but other options exist.

b) Add a keypad to the system so that you can use it to play tones. Make one of the keys play the melody.

Task 4 – Number representation (1+1p)

a) Do a bitwise not on a positive integer (`int`). What is the resulting number? What if you do the same for a unsigned integer (`unsigned int`)? Explain how the microcontroller stores negative numbers on bit level.

Hint: Printing the integer in different formats may help. See Serial.println() in the Arduino language reference.

b) Why doesn't the following code work as one might expect? How would you make the comparison work with floating point numbers? Modify the code so that it gives the expected result. Do not change the first line of code.

```
float f = pow(sqrt(2),2);
Serial.println(f);

if(f == 2.0){
    Serial.println("True");
}else{
    Serial.println("Why do we end up here?");
}
```

Hint: Language Reference or http://tito-perusteet-2022.mooc.fi/luku-3/2-kokonaisluvut-ja-liukuluvut. An excellent floating point converter can be found here (h-schmidt.net).

> There are many ways to achieve the described functionalities but in tasks 1 and 2 we want you to familiarize yourself with interrupts and timers. These enable faster response times and free processor resources in more demanding applications, such as your final project.