Exercise 1: Use a LED for Morse code

Equipment

For this exercise you will need:

- 1 x Arduino Uno
- 1 x LED
- 1 x Resistor ($\sim 60\Omega - 220\Omega$)
- Wires

Remember to select your USB port.
Tools→Port

Reading

Chapter 2 - 5

Setup

Open the "Blink" program for inspiration (In Arduino IDE: File → Examples → 01.Basics → Blink).

- Connect the LED to any digital pin on the Uno. Use a resistor to limit the current going through the LED.
- Use pinmode(PIN, OUTPUT) to setup the pin for output (Change PIN for the name of the digital pin you picked).
- Define your constants
  const uint8_t ledPin = <value>; //e.g. D6
  const int timeUnit = <value>; //e.g. 500

Questions & Exercises

1a: Morse code uses 5 "bits" for representing numbers 0-9, how many bits are needed if you would use binary? Why is this not possible in Morse code?
1b: What is the value of **a** after the loop?

```
int a = 1;
for(int i = 0; i < 5; i++){
    a += a;
}
```

Now it is time to make a program that executes Morse Code using the LED.
1c: Make a program that morses "SOS". You should follow the international requirement for morse code shown in figure 1.
1d: Try setting ledPin = LED_BUILTIN, what happens?
1e: Update the program to morse your name
1f: Finally, update the program to morse your name using for-loops

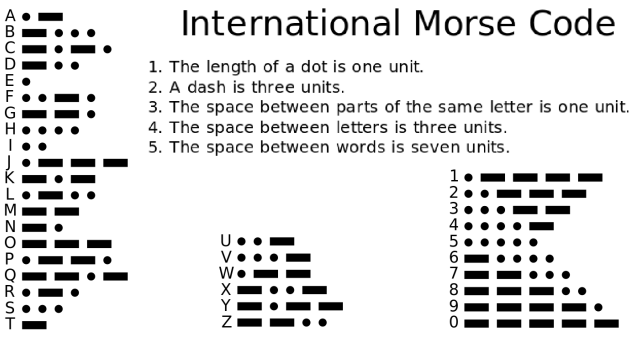


Figure 1: Morse code

FOR-loops

For-loops are useful for repeating something a known number of times.

```
for(int i = 0; i < 10; i++){
 //this loop will run ten times
 }
```

Hint

Use digitalWrite(ledPin, HIGH) to turn on the LED the wait some time with delay(timeUnit) do the same to turn a LED off, just set the pin LOW instead of HIGH

Exercise 2: More LEDs

Equipment

For this exercise you will need:

- 1 x Arduino Uno
- 1 x Red LED
- 1 x Green LED
- 1 x Yellow LED
- 3 x $\sim 60 - 220\Omega$ Resistors
- Wires
- *Optional: 2 x Red LED*

> The **serial monitor** allows you to read the printouts from Serial.print(). Open it using **ctrl+shift+m** or by clicking on the magnifying glass in the top right corner.

Reading

Chapter 2 - 5

Setup

- Connect the anodes of the LEDs to 3 different digital ports on the Uno. Use resistors to limit the current going through the LEDs.
- Set the baudrate of the communication to 115200 $\frac{bit}{s}$ with Serial.begin(115200) in setup(){. . .}

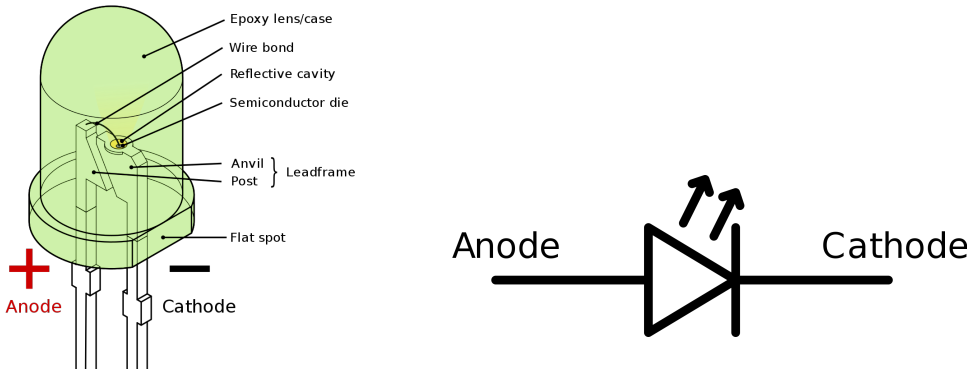


Figure 2: LED

Questions & Exercises

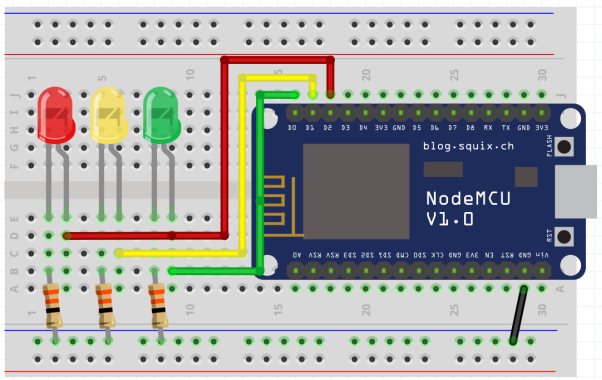
2a: You should be used to the normal arithmetic operators + - * / , but do you know this operator: %? What is 42 % 5?
2b: Make a traffic light:

- Implement the blinking sequence of a traffic light. (no sensor, just the light)
- Use Serial.print() to write an instruction for the drivers every time the light changes, e.g. "STOP!"

2c: Make a binary counter:

- *Optional: change all the LEDs to red*
- Use the three LEDs to show the value of the counter in binary.
- The counter should count from 0 (000_{BIN}) to 7 (111_{BIN}) and then start over.
- Print the value of the counter to the serial monitor.

2d: What could you have used the %-operator for in this exercise?



Hint

You may want to write functions to help you out e.g.

```
void YellowGreen(){
    DigitalWrite(D5, HIGH);
    DigitalWrite(D6, HIGH);
    delay(2000);
    DigitalWrite(D5, LOW);
    DigitalWrite(D6, LOW);
}
```

Exercise 3: Digital Input

Equipment

For this exercise you will need:

- 1 x Arduino Uno
- 1 x LED
- 1 x Button
- 1 x Resistor $\sim 60 - 220\Omega$
- Wires

Use **Tab** to add indentation to all selected lines

Reading

Chaptor 3 & 4

Setup

- Connect the LED and the button to the Uno. The button should be connected to ground and to one of the digital pins. See figure 3
- Remember to set the pin to input using pinMode(PIN, INPUT_PULLUP)

Questions & Exercises

3a: What is the difference between pinMode(PIN, INPUT) and pinMode(PIN, INPUT_PULLUP)?
3b: What is the operator ! used for?
3c: Control the LED with the button

1. While the button is pushed down the LED should be turned on.
   You can use digitalRead(PIN) and digitalWrite(PIN, <value>) to read from the button pin and write to the LED pin
2. Make a latching button. A latching button should change state if you push it. (Push to turn on LED. Push again to turn LED off)

. **3d:** How often does your program check if the button has been pushed? Does this seem reasonable?

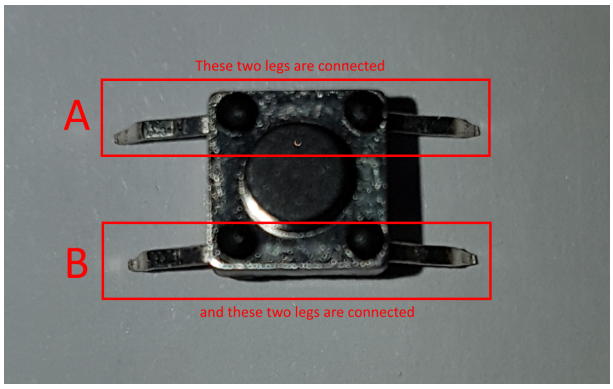


Figure 3: Button layout: The two pins in each pair is essentially the same pin. One pair should be connected to ground and the other to a digital input pin

---

Hint

```
pinMode(b1, INPUT_PULLUP); // Setup button pin
if(digitalRead(b1) == false){
    //button pushed
}
```

Exercise 4: Fritzing

Equipment

For this exercise you will need:

- 1 x Circuit from exercise 3
- Fritzing software

Setup

- Download and install Fritzing
- Tidy up your circuit.

Exercise

1. Draw your circuit using Fritzing.

.

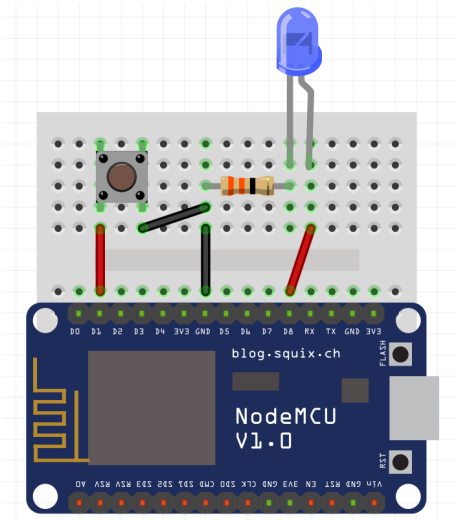


Figure 4: A circuit drawing in Fritzing.

Hint

Always document your work, you might forget something :)