ARDUINO

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists and anyone interested in creating interactive objects or environments.

Shields: Are add-ons which can expand the functionality with Ethernet, Wi-Fi, Bluetooth, sensors, LCD-screens, buttons, motor control etc.

AN EXAMPLE ARDUINO - SKETCH:

```
// set pin numbers:
int buttonPin = 2;     // the number of the pushbutton pin
int ledPin =  13;      // the number of the LED pin

int buttonState = 0;   // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

COMMENTS ARE IGNORED

DON'T FORGET!

SETUP IS RUN ONLY ONCE

LOOP RUNS CONTINUOUSLY

ASSIGNMENT

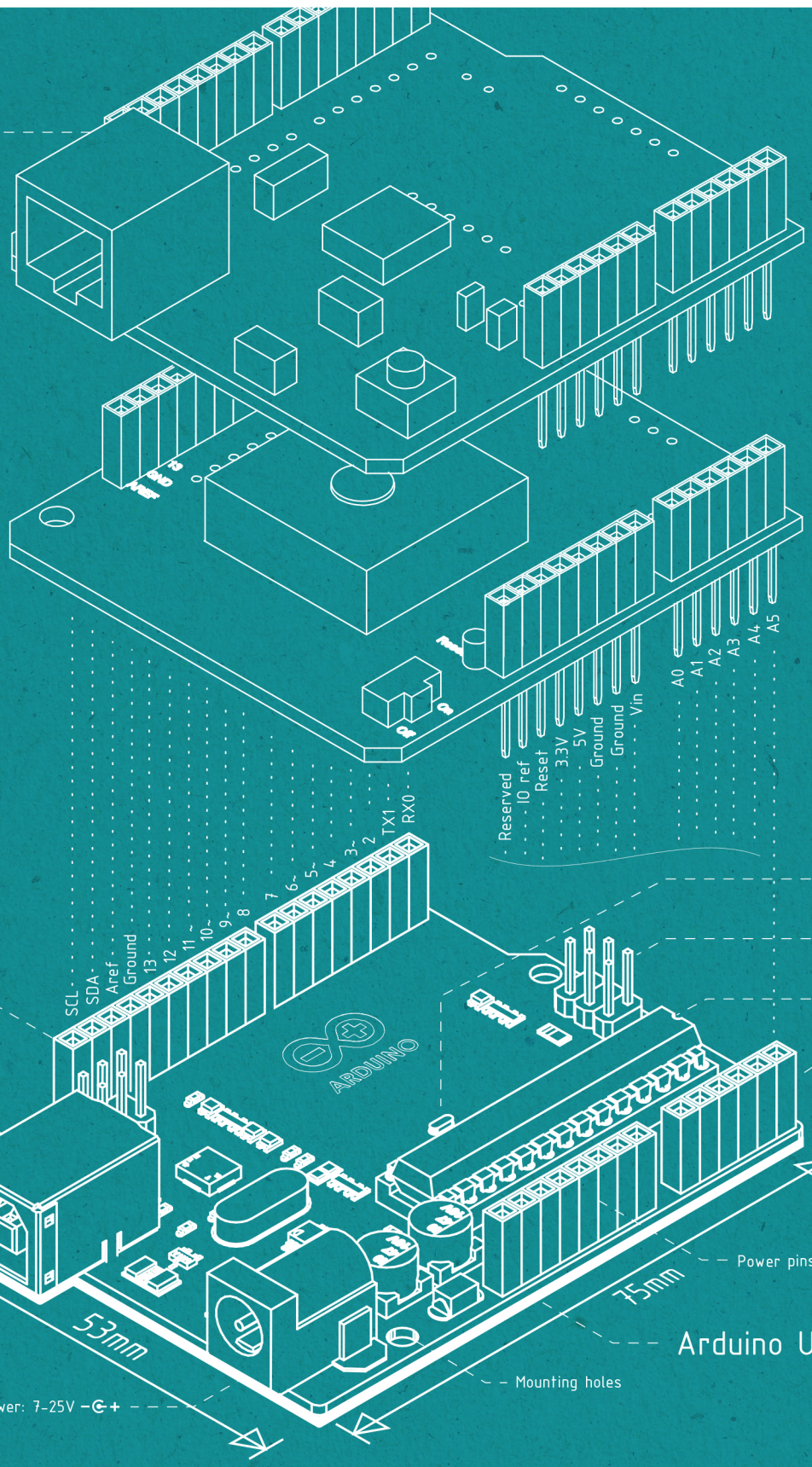
COMPARISON:
`==` EQUAL
`!=` NOT EQUAL
`<` LESS THAN
`>` GREATER THAN
`<=` LESS OR EQUAL TO
`>=` GREATER OR EQUAL TO
`&&` AND
`||` OR
`!` NOT

=1

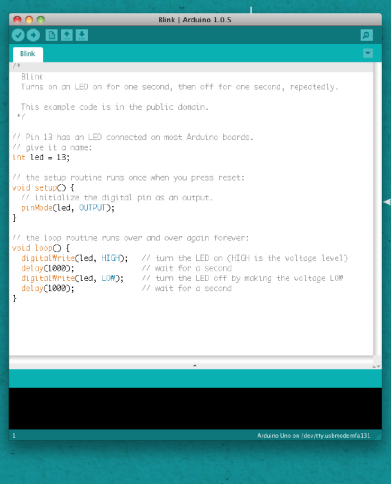
=Ø

ALSO DON'T FORGET!

```
// select the input pin for the potentiometer
int sensorPin = A0;
// variable to store the value coming from the sensor
int sensorValue = 0;
//Read a value (0-1023) from "sensorPin"
sensorValue = analogRead(sensorPin);
```

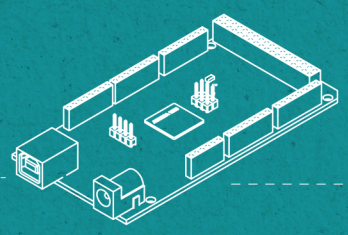


Software: The Arduino IDE is a simple to use programming environment. It allows a user to write code, to test & compile and to upload the program into the microcontroller on an Arduino platform.

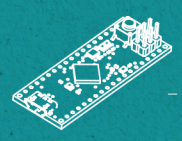



```
// Useful functions:
+ - / *                                  // arithmetic
delay(1000)                              // wait for one second (1000 milliseconds)
millis()                                 // returns the time since start
constrain(value, low, high)              // constrains a value
map(value, fromLow, fromHigh, toLow, toHigh)  // Re-maps a number from one range to another
Serial.begin(9600)                       // Opens a serial port with a 9600 baudrate
Serial.print()                           // Writes human-readable text to the serial port
Serial.write()                           // More advanced alternative
Serial.read()                            // Read one byte from the serial port
```

Arduino Open-Source Boards come in various forms, made by Arduino and others supporting the community.



Arduino Mega 2560
More pins and speed



Arduino Micro
Smaller size

Arduino Lilypad
For wearable electronics

Overview

An Arduino is an open-source electronics prototyping platform, running an Atmel ATMEGA microcontroller, which can sense the environment through sensors (light, sound, pressure, force, voltage, etc.) and can change its environment through actuators (motors, lights, etc.). It can also communicate with the outside world through USB, and, with the addition of shields, over Ethernet, Wi-fi, GSM, and a lot of other options. It is programmed with the Arduino language and the Arduino Development Environment (IDE).

Designers, tinkerers and artists have used the platform to create numerous interactive installations, measurement devices, internet-controlled cat feeders and much more. Over the years, the Arduino platform has grown to become a building block of the most open-source hardware projects and an ecosystem of developers, designers and sellers emerged.

Community

One of the key strengths of the Arduino platform is the existance of a very large and vibrant community. These hobbyists and professionals support each other and develop new hardware and software. This means that it's very easy to get started developing something for the average user; whatever you're trying to build, someone else probably already worked on that before you and shared their work.

Shields & Libraries

Over the years, both Arduino (the company) and third parties designed hundereds of shields for various purposes. These PCB's can be stacked on top of the Arduino board. There are shields to connect to networks (Ethernet, Wi-Fi, Bluetooth, GSM), to control things (servo's, motors, lights, electrical appliances), to interact with users (screens, buttons, audio, LED's), and to store information, provide power, read RFID-cards etc. Usually, a shield is provided with a software library which makes it easy to write software for the hardware in the shield. This way, a user can simply plug in the shield, load a library and get started with his program without having to think about the complex details of interfacing with e.g. an LCD-screen.

Hardware

The Arduino Uno board has:

- 14 digital inputs and outputs (IO). These are used to interact with buttons, LED's etc. Of these, 6 have PWM "analog output" capability which are marked with ~. Can source (supply) or sink (receive) a maximum of 40mA per pin, 200mA for all pins combined.
- 6 analog inputs. Mostly used to read analog sensors such as potentiometers, temperature sensors, and force sensors. Can also be used as digital IO.
- USB connection (B-type female), to provide power (5V, 500mA), programming and a serial data connection up to 115200 baud.
- Barrel power jack, to provide power between 7-12V from an adapter or battery. This is regulated on the board to 5V (200mA) and 3.3V (50mA). The board automatically switches between power from USB or barrel.
- ICSP header, to program the bootloader and as an alternative programming method.
- 16MHz crystal, to provide a steady clock signal to the microcontroller.
- Reset button, which restarts the microcontroller and the software.
- A small LED, connected to pin 13, for testing and debugging.
- 32KB of flash memory. Contains the "sketch" that you program and the bootloader for programming.
- 2KB of SRAM. Used as the working memory to perform functions of the software.
- 1KB of EEPROM. Used as long term (parameter) storage, which "survives" programming.

Several protocols are available to communicate with peripherals, such as UART (also used for programming, pin 0 and 1), SPI (pin 11, 12, 13), Two Wire, One Wire, I2C.

GENERATORELECTRONICS.INFO
YOURDUINO.COM

element14 COMMUNITY
CIRCUIT CELLAR THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION
elektor
interactive objects