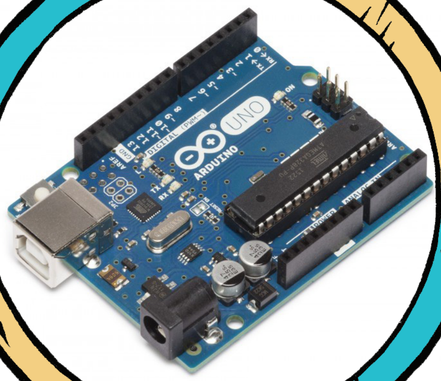
TOOL

RELEVANT CARDS

analog inputs



MEASURING A VOLTAGE

How to measure a voltage precisely
with analog inputs

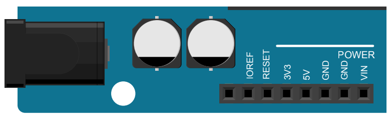
Measuring range: 0 to 5 V
Accuracy: 5 mV default
Ease of use: simple
Usage: use Arduino as a voltmeter

By default, the analog inputs can be considered as voltmeters, operating between 0 and 5 volts, with an accuracy of 5 mV. These inputs are connected to an analog-to-digital converter (ADC) which linearly converts the input voltage into a number between 0 (0 volts) and 1023 (5 volts).

```
float Vmes ;  // decimal variable
Vmes = analogRead(A0) * 5.0 / 1023.0 ; // measured voltage, in volts
```

Stability of the reference

During the process of measuring a voltage and converting it into a number, the ADC compares this voltage to a reference voltage of 5 V. If the Arduino board is powered only by a USB cable, the reference voltage is provided by this cable. This voltage is not necessarily very precise, nor very stable: the accuracy of measurements and their stability can be affected.



Powering the Arduino board through an external source other than the USB cable improves the stability of the 5 V reference. The power can be supplied either via the dedicated jack port (recommended) or via the Vin port. It must be a voltage between 6 and 12 V. The reference 5 V used by the ADC is then produced by a voltage regulator included on the board. This solution improves the stability of the reference voltage, but the absolute value is not necessarily exactly 5 V: depending on the boards, the value can vary between 4.8 and 5.2 V (but for a given board, this value remains stable). For accurate measurements, it is best to calibrate this reference voltage, by comparing a measurement made by the Arduino board to that of another device, for example.

TOOL – MEASURING A VOLTAGE

Changing the reference

By default, the DAC works from 0 to 5 volts. If the voltage to be measured varies over a smaller range, for example from 0 to 1 volt, the numbers returned by the DAC will be between 0 and 102, and the higher numbers will never be used. In such a case, changing the DAC reference voltage can improve the precision.

The reference voltage of the DAC is the voltage that is converted to the value 1023. This voltage must be positive, and the value 0 always corresponds to 0 V, the voltage of the port GND. One way to change the reference voltage is to use the **analogReference ()** instruction. Here is an example for the Uno card (for other cards, check on the arduino.cc reference website):

```
void setup() {
  analogReference(INTERNAL) ;
}
```

The reference voltage is now 1.1 volts (another board reference voltage).

```
void loop() {
  float Vmes ;
  Vmes = analogRead(A0) * 1.1 / 1023.0 ;
}
```

new conversion formula:
1023 = 1,1 V

Note: the 1.1 V board voltage regulator, just like the 5 V regulator, is stable (its voltage does not change over time), but not necessarily very accurate: its voltage is between 1.0 and 1.2 V, and varies from board to board. For accurate measurements, calibrate this voltage.

Note: An external voltage can also be used as reference for the DAC. To do so, it is necessary to connect this voltage on the Aref port, and to use in the setup the instruction **analogReference (EXTERNAL)** . The conversion formula must be modified accordingly.

Averaging measurements

Averaging improves the measuremnt accuracy only if the measurements fluctuate over time. If the measurements always give the same number, the result of averaging will be identical.

The memory of the Uno board is not very large, special care should be taken on how to use it. The most efficient way is to do as many calculations as possible with integers, and to use the decimal numbers as the last stage (these numbers require more memory space).

```
unsigned long Sum = 0 ;     // unsigned longinteger variable
int N = 1000 ;              // Number of measurements to average
float Vmes ;

for(int i = 0 ; i < N ; i++) {      // this sequence is executed N times
  Sum = Sum + analogRead(A0) ;     // summation of the measurements
}
Vmes = Sum / N * 5.0 / 1023.0 ;
```