unsigned int

[Data Types]

Description

On the Uno and other ATMEGA based boards, unsigned ints (unsigned integers) are the same as ints in that they store a 2 byte value. Instead of storing negative numbers however they only store positive values, yielding a useful range of 0 to 65,535 ((2^16) - 1).

The Due stores a 4 byte (32-bit) value, ranging from 0 to 4,294,967,295 (2^32 - 1).

The difference between unsigned ints and (signed) ints, lies in the way the highest bit, sometimes referred to as the "sign" bit, is interpreted. In the Arduino int type (which is signed), if the high bit is a "1", the number is interpreted as a negative number, and the other 15 bits are interpreted with ([2’s complement math](http://en.wikipedia.org/wiki/2%27s_complement)).

Syntax

unsigned int var = val;

Parameters

var: variable name.  
val: the value you assign to that variable.

Example Code

unsigned int ledPin = 13;

Notes and Warnings

When unsigned variables are made to exceed their maximum capacity they "roll over" back to 0, and also the other way around:

unsigned int x;

x = 0;

x = x - 1; // x now contains 65535 - rolls over in neg direction

x = x + 1; // x now contains 0 - rolls over

Math with unsigned variables may produce unexpected results, even if your unsigned variable never rolls over.

The MCU applies the following rules:

The calculation is done in the scope of the destination variable. E.g. if the destination variable is signed, it will do signed math, even if both input variables are unsigned.

However with a calculation which requires an intermediate result, the scope of the intermediate result is unspecified by the code. In this case, the MCU will do unsigned math for the intermediate result, because both inputs are unsigned!

unsigned int x = 5;

unsigned int y = 10;

int result;

result = x - y; // 5 - 10 = -5, as expected

result = (x - y) / 2; // 5 - 10 in unsigned math is 65530! 65530/2 = 32765

// solution: use signed variables, or do the calculation step by step.

result = x - y; // 5 - 10 = -5, as expected

result = result / 2; // -5/2 = -2 (only integer math, decimal places are dropped)

Why use unsigned variables at all?

* The rollover behaviour is desired, e.g. counters
* The signed variable is a bit too small, but you want to avoid the memory and speed loss of long/float.