

## CoCoALib - Support #1666

### MachineInt: chase through ULL changes

16 Feb 2022 19:39 - John Abbott

<b>Status:</b>	In Progress	<b>Start date:</b>	16 Feb 2022
<b>Priority:</b>	Normal	<b>Due date:</b>	
<b>Assignee:</b>		<b>% Done:</b>	10%
<b>Category:</b>	Tidying	<b>Estimated time:</b>	0.00 hour
<b>Target version:</b>	CoCoALib-0.99900	<b>Spent time:</b>	0.85 hour
<b>Description</b> (2022-02-16) now using ULL as internal repr for MachineInt. Must chase though all changes: <ul style="list-style-type: none"><li>• check all calls to AsSignedLong and/or AsUnsignedLong</li><li>• check all calls to uabs (change to ul_abs or ull_abs)</li></ul>			
<b>Related issues:</b>			
Related to CoCoALib - Design #581: C++14: MachineInt		<b>Closed</b>	<b>04 Jul 2014</b>
Related to CoCoALib - Design #1804: Use long long (at least sometimes)?		<b>In Progress</b>	<b>25 Mar 2024</b>
Related to CoCoALib - Design #934: MachineInt: change semantics?		<b>In Progress</b>	<b>30 Sep 2016</b>
Related to CoCoALib - Feature #828: MachineInt: function for checking that va...		<b>In Progress</b>	<b>30 Nov 2015</b>

#### History

##### #1 - 16 Feb 2022 19:39 - John Abbott

- Related to Design #581: C++14: MachineInt added

##### #2 - 16 Feb 2022 19:42 - John Abbott

We will need new tests too.

Oh joy! That C++ has inherited so many "complicated/confusing" integer types...  
Designed for speed rather than safety.

I had also hoped that BOOST's `numeric_cast` would have been adopted in the STL... not yet, it seems.

##### #3 - 15 Feb 2024 22:40 - John Abbott

- Target version changed from CoCoALib-0.99850 to CoCoALib-0.99900

##### #4 - 15 Apr 2024 10:03 - John Abbott

- Related to Design #1804: Use long long (at least sometimes)? added

##### #5 - 15 Apr 2024 10:05 - John Abbott

I wonder if this change should be reconsidered in light of the discussion in [#1804](#)?

##### #6 - 15 Apr 2024 10:06 - John Abbott

- Related to Design #934: MachineInt: change semantics? added

##### #7 - 15 Apr 2024 10:08 - John Abbott

- Related to Feature #828: MachineInt: function for checking that value is greater than some lower limit (and below MAXLONG) added

**#8 - 16 Apr 2024 22:40 - John Abbott**

- Status changed from New to In Progress

- % Done changed from 0 to 10

I have just removed long long from MachineInt (and had to change one call in BigIntOps.C)

There are actually **2** implementations of MachineInt: one uses an internal repr wwhich is unsigned long and a bool to say whether the value was negative; the other uses signed long without any extra bool [I have not yet updated this alternative impl]

The alternative impl is "neater" (just 1 data field) and likely faster; but is such speed important if the value is anyway going to be converted to a RingElem (or used in an operation with a RingElem)?