Ruby - Bug #21436

Date#ajd returns incorrect positive values due to integer overflow for large negative years

06/11/2025 12:27 PM - Stranger6667 (Dmitry Dygalo)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	3.2.8 - 3.4.4 (and older versions too)	**Backport:**	3.2: UNKNOWN, 3.3: UNKNOWN, 3.4: UNKNOWN

Description

Date#ajd (astronomical Julian Day number) returns incorrect positive values instead of negative values for certain large negative years due to signed integer overflow in the RB_INT2FIX function.

```
require 'date'

# Works correctly
Date.civil(-110823815392979, 2, 19).ajd
#=> (-80956797141128945/2)

# BUG - returns positive instead of negative
Date.civil(-11082381539297990, 2, 19).ajd
#=> (112769232240103 6327/2)  # Should be negative!

# Works correctly (takes a different code path)
Date.civil(-11111111108238153929 7990, 2, 19).ajd
#=> (-81166666645679714453739481/2)
```

In ext/date/date_core.c, the ir value can overflow

```
if (FIXNUM_P(r) && FIX2LONG(r) <= (FIXNUM_MAX / 2)) {
    long ir = FIX2LONG(r);
    ir = ir * 2 - 1;
    // On Date.civil(-11082381539297990, 2, 19).ajd
    //
    // ir = -8095679714453739481
    return rb_rational_new2(LONG2FIX(ir), INT2FIX(2));  // Overflow in LONG2FIX
}
```

Then in include/ruby/internal/arithmetic/long.h (LONG2FIX is an alias for RB_INT2FIX)

```
static inline VALUE
RB_INT2FIX(long i)
{
    RBIMPL_ASSERT_OR_ASSUME(RB_FIXABLE(i));

    /* :NOTE: VALUE can be wider than long.  As j being unsigned, 2j+1 is fully
     * defined. Also it can be compiled into a single LEA instruction. */
    const unsigned long j = RBIMPL_CAST((unsigned long)i);
    const unsigned long k = (j << 1) + RUBY_FIXNUM_FLAG;
    const long          l = RBIMPL_CAST((long)k);
    const SIGNED_VALUE  m = l; /* Sign extend */
    const VALUE         n = RBIMPL_CAST((VALUE)m);

    RBIMPL_ASSERT_OR_ASSUME(RB_FIXNUM_P(n));
    return n;
}
```

It effectively goes:

```
j = (unsigned long)-8095679714453739481  // 10351064359255812135
k = (10351064359255812135 << 1) + RUBY_FIXNUM_FLAG // 2255384644802072655
```

```
...
```

So, 2255384644802072655 encodes 1127692322401036327, which is the numerator for the observed ajd return value in the reproducer above.

It also affects Date comparisons (since ajd is used in cmp_gen)

Associated revisions

Revision 022c18b6 - 06/15/2025 04:12 PM - Dmitry Dygalo

[ruby/date] [Bug #21436] check for fixnum lower bound in m_ajd

Issue - https://bugs.ruby-lang.org/issues/21436

Apparently, the lower bound check is missing, which results in overflow & wrapping later on in RB_INT2FIX

Signed-off-by: Dmitry Dygalo dmitry.dygalo@workato.com

https://github.com/ruby/date/commit/67d75e8423

History

#1 - 06/15/2025 04:24 PM - Anonymous

- Status changed from Open to Closed

Applied in changeset git|022c18b60d2245980abcdd7b5195eebca73b8809.

[ruby/date] [Bug #21436] check for fixnum lower bound in m_ajd

Issue - https://bugs.ruby-lang.org/issues/21436

Apparently, the lower bound check is missing, which results in overflow & wrapping later on in RB_INT2FIX

Signed-off-by: Dmitry Dygalo dmitry.dygalo@workato.com

https://github.com/ruby/date/commit/67d75e8423