

Re: New and faster algorithm for multiplication

Source: <http://sci.tech-archive.net/Archive/sci.math/2005-09/msg01392.html>

- *From:* "Proginoskes" <CHeckman@xxxxxxxxxx>
 - *Date:* 6 Sep 2005 23:27:58 -0700
-

Hans Petter Selasky wrote:

> Hi,
>
> I think I have found a new and faster algorithm for multiplication.
>
> Any comments?

What's the running time? If you have two N-bit integers, how long does it take to find their product?

--- Christopher Heckman

> Inlined attachment "multiply.c":
>
> /*-
> * Copyright (c) 2005 Hans Petter Selasky. All rights reserved.
> *
> * Redistribution and use in source and binary forms, with or without
> * modification, are permitted provided that the following conditions
> * are met:
> * 1. Redistributions of source code must retain the above copyright
> * notice, this list of conditions and the following disclaimer.
> * 2. Redistributions in binary form must reproduce the above copyright
> * notice, this list of conditions and the following disclaimer in the
> * documentation and/or other materials provided with the distribution.
> *
> * THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
> * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
> * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
> PURPOSE
> * ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
> * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
> CONSEQUENTIAL
> * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
> * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
> * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
> STRICT
> * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
> * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF

Re: New and faster algorithm for multiplication

```
> * SUCH DAMAGE.
> */
> #include <stdio.h>
>
> /*
> * NOTE: "unsigned" is 32-bit
> */
>
> static unsigned
> multiply(unsigned a, unsigned b, unsigned temp)
> {
> unsigned carry = 0;
> unsigned old1;
> unsigned old2;
> unsigned char n = 32;
>
> b ^= (2*b);
>
> /* this subtraction can be optimized if the
> * multiplication factor is a constant:
> */
> a -= 1;
>
> while(n-->
> {
> /* as you can see, there is
> * no addition circuit that
> * must be waited for,
> * and consequently
> * this loop can proceed
> * very quickly !
> */
>
> if(b & 1)
> {
> old1 = temp;
> old2 = a;
> }
> else
> {
> old1 = carry;
> old2 = 0;
> }
>
> temp ^= (carry | old2);
> carry = temp & old1;
>
> carry *= 2;
> a *= 2;
> a |= 1;
> b /= 2;
```

Re: New and faster algorithm for multiplication

```
> }
> return temp;
> }
>
> int main()
> {
> unsigned x,y,z,t,u;
>
> /* verify that it works */
>
> for(x = 0; x < 64; x++)
> {
> for(y = 0; y < 64; y++)
> {
> for(z = 0; z < 64; z++)
> {
> t = multiply(x,y,z);
> u = ((x*y)+z);
>
> if(t != u)
> {
> printf("error: (((%d*%d) + %d) = %d) != %d\n",
> x,y,z,u,t);
> }
> }
> }
> }
> return 0;
> }
```

- **Follow-Ups:**

- ◆ **[Re: New and faster algorithm for multiplication](#)**

- ◆ *From:* Hans Petter Selasky

- **References:**

- ◆ **[New and faster algorithm for multiplication](#)**

- ◆ *From:* Hans Petter Selasky

- Prev by Date: **[Re: infinity](#)**

- Next by Date: **[Re: Where do mathematical ideas come from?](#)**

- Previous by thread: **[Re: New and faster algorithm for multiplication](#)**

- Next by thread: **[Re: New and faster algorithm for multiplication](#)**

- Index(es):

- ◆ **[Date](#)**

- ◆ **[Thread](#)**