Fast modular exponentiation

Case $e = 2^k$

Suppose that we need to compute b to the power 2^k modulo m using only k modular multiplications (without using explicit exponentiation function). Denote $b_i := b^{2^i} \bmod m$.

- $b_0 = b \bmod m$.
- $b_1 = b_0 \times b_0 \bmod m$. As we wanted, $b_1 = b^2 \bmod m$.
- $b_2 = b_1 \times b_1 \bmod m$. Since $b_1 = b^{2^1} \bmod m$, we have $b_1 \times b_1 = b^2 \times b^2 = b^{2+2} = b^4 \bmod m$.
- $b_3 = b_2 \times b_2 \bmod m$. Again, since $b_2 = b^4 \bmod m$, we have $b_3 = b^4 \times b^4 = b^8 \bmod m$.
  ...
- $b_k = b_{k-1} \times b_{k-1} \bmod m$. Since $b_{k-1} = b^{2^{k-1}} \bmod m$, we have $b_k = b^{2^{k-1}} \times b^{2^{k-1}} = b^{(2^{k-1}+2^{k-1})} = b^{2^k} \bmod m$.

Our algorithm consists of k steps, on each step we perform only one modular multiplication. Note that since all numbers $b_1, b_2, \ldots, b_k$ are remainders modulo m, we don't need to multiply long numbers.

Arbitrary e

Let $e_n e_{n-1} \ldots e_1 e_0$ be the binary representation of e. By definition,

$$e = e_n \times 2^n + e_{n-1} \times 2^{n-1} + \ldots + e_1 \times 2^1 + e_0 \times 2^0,$$

thus

$$b^e = b^{((e_n \times 2^n)+(e_{n-1} \times 2^{n-1})+\ldots+(e_1 \times 2^1)+(e_0 \times 2^0))} = b^{e_n \times 2^n} \times b^{e_{n-1} \times 2^{n-1}} \times \ldots \times b^{e_1 \times 2^1} \times b^{e_0 \times 2^0}.$$

Note that for all i, $e_i \in \{0, 1\}$, therefore, if $e_i = 1 : \ b^{e_i \times 2^i} = b^{2^i}$, if $e_i = 0 : \ b^{e_i \times 2^i} = 1$. Hence, b^e is the product of b^{2^i} for all i such that $e_i = 1 \iff e_i \neq 0$. Therefore, the fast modular exponentiation algorithm for arbitrary e works as follows:

1. Compute $e_n e_{n-1} \ldots e_1 e_0$ — the binary representation of e.
2. Compute b^{2^i} for all $i = 0, 1, 2, \ldots n$ — we have already have an efficient algorithm for that!
3. Multiply together b^{2^i} for all i such that $e_i = 1$. (NB: After each multiplication take modulo m to avoid long numbers!)