Formal Analysis and Proof of the `NMOD_RED2` Reduction Algorithm

1 Introduction and Setup

Let $W = b$ be the machine word size in bits, and let the radix be $B = 2^b$. We are given a two-word dividend $x = u_1 B + u_0$ where $0 \le u_1, u_0 < B$, and a pre-normalized divisor n such that $B/2 \le n < B$. The goal is to compute $x \bmod n$.

Let the divisor be expressed as $n = B/2 + k$, for some integer $0 \le k < B/2$.

2 Precomputation of the Inverse

The algorithm relies on a precomputed Möller-Granlund pseudo-inverse v, defined mathematically as:

$$v = \left\lfloor \frac{B^2 - 1}{n} \right\rfloor - B$$

In the C code, this is implemented using a hardware 2-by-1 limb division that avoids overflow:

```
udiv_qrnnd(ninv, r, ~n, ~UWORD(0), n);
```

Mathematically, the bitwise NOT operations construct a dividend $D = (B - 1 - n)B + (B - 1) = B^2 - 1 - nB$. The division computes:

$$v = \left\lfloor \frac{B^2 - 1 - nB}{n} \right\rfloor = \left\lfloor \frac{B^2 - 1}{n} \right\rfloor - B$$

Lemma 1. *If $k \ge 1$ and $16k^2 \le B$, then $v = B - 4k$.*

Proof. We can expand the numerator $B^2 - 1$ algebraically in terms of $n = B/2 + k$:

$$B^2 - 1 = (B/2 + k)(2B - 4k) + 4k^2 - 1 = n(2B - 4k) + 4k^2 - 1$$

Since $k \ge 1$, the remainder term $4k^2 - 1 \ge 3 \ge 0$. By the condition $16k^2 \le B$, we have $4k^2 \le B/4$. Thus, $4k^2 - 1 < B/4 < B/2 \le n$. Because the remainder is strictly between 0 and $n - 1$, the integer quotient is exactly $2B - 4k$. Therefore, $v = (2B - 4k) - B = B - 4k$.
(Note: If $k = 0$, $n = B/2$ and $v = B - 1$. We proceed assuming $k \ge 1$, as $k = 0$ is a trivial case.) □

3 The `NMOD_RED2` Algorithm

The core algorithm calculates a quotient estimate q_1 and a remainder estimate.

1. Multiply and Add: Compute the 2-limb product $P = u_1 v + u_1 B + u_0$.

2. Extract Words: Split P into a high word q_1 and low word q_0 such that $P = q_1B + q_0$.

3. Initial Remainder: $r_1 \equiv u_0 - (q_1 + 1)n \pmod{B}$.

4. Correction 1: `if` $(r_1 > q_0)$ then $r_1 \leftarrow (r_1 + n) \pmod{B}$.

5. Correction 2: `if` $(r_1 < n)$ return r_1, `else` return $r_1 - n$.

4 The Conjecture

Conjecture 2. *The `NMOD_RED2` algorithm correctly computes $x \bmod n$ for all $0 \le x < B^2$ provided that:*

$$B/2 \le n \le B/2 + \lfloor \sqrt{2^{b-4}} \rfloor$$

Because $B = 2^b$, the upper bound implies $k \le \lfloor \sqrt{B/16} \rfloor$, which fundamentally guarantees that $16k^2 \le B$.

5 Formal Proof

5.1 Quotient Estimation

Using Lemma 1, we substitute $v = B - 4k$ into the product P:

$$P = u_1(B - 4k) + u_1B + u_0 = 2u_1B - 4ku_1 + u_0$$

We define an integer γ to handle the fractional carry over the radix B:

$$\gamma = \left\lceil \frac{4ku_1 - u_0}{B} \right\rceil$$

Because $0 \le u_1 < B$ and $0 \le u_0 < B$, we can bound γ:

$$0 = \left\lceil \frac{0 - (B-1)}{B} \right\rceil \le \gamma \le \left\lceil \frac{4k(B-1) - 0}{B} \right\rceil \le 4k$$

Using γ, we formally define the upper and lower words of $P = q_1B + q_0$:

$$q_1 = 2u_1 - \gamma \tag{1}$$
$$q_0 = u_0 - 4ku_1 + \gamma B \tag{2}$$

where by the definition of γ, we have $0 \le q_0 < B$.

5.2 Bounding the True Estimated Remainder

Let r^* be the mathematical remainder based on our quotient estimate before any modulo B truncation:

$$r^* = x - q_1n = (u_1B + u_0) - (2u_1 - \gamma)(B/2 + k)$$

Expanding this expression:

$$r^* = u_1B + u_0 - u_1B - 2ku_1 + \gamma B/2 + \gamma k = u_0 - 2ku_1 + \gamma B/2 + \gamma k$$

From (2), we know $2ku_1 = (u_0 - q_0 + \gamma B)/2$. Substituting this into r^*:

$$r^* = u_0 - \frac{u_0 - q_0 + \gamma B}{2} + \frac{\gamma B}{2} + \gamma k = \frac{u_0 + q_0}{2} + \gamma k$$

Because $u_0, q_0 \leq B - 1$ and $\gamma \leq 4k$, the maximum value of r^* is strictly bounded:

$$r^* \leq (B - 1) + 4k^2 < B + 4k^2$$

By the conjecture's condition $16k^2 \leq B$, we know $4k^2 \leq B/4$. Therefore:

$$r^* < 1.25B$$

5.3 Algorithm Correction and Modulo Arithmetic

The algorithm calculates $r_1 \equiv u_0 - (q_1 + 1)n \equiv x - q_1 n - n \equiv r^* - n \pmod{B}$. We analyze the three possible error states of the quotient estimate q_1:

Case 0: $r^* < n$ (Quotient q_1 is correct). The algorithm over-subtracted. Here, $-B < -n \leq r^* - n < 0$. Under modulo B, $r_1 = B + r^* - n$. For the algorithm to recover, the check $(r_1 > q_0)$ must be TRUE.

$$B + \frac{u_0 + q_0}{2} + \gamma k - B/2 - k > q_0 \implies B/2 + \frac{u_0 - q_0}{2} + k(\gamma - 1) > 0$$

If $\gamma \geq 1$, $k(\gamma - 1) \geq 0$, and since $B/2 + (u_0 - q_0)/2 > 0$, the inequality holds. If $\gamma = 0$, then $4ku_1 \leq u_0$. Thus $q_0 = u_0 - 4ku_1$, which means $u_0 - q_0 = 4ku_1 \geq 0$. The inequality becomes $B/2 + 2ku_1 - k > 0$. Since $B \geq 16$ and $k \geq 1$, $B/2 \geq 8k^2 > k$, making the sum strictly positive. Thus, $(r_1 > q_0)$ is always true. The algorithm adds n, yielding $r_2 = B + r^* \equiv r^* \pmod{B}$. Since $r^* < n$, the final check correctly returns r^*.

Case 1: $n \leq r^* < 2n$ (Quotient q_1 is off by 1). Here, $0 \leq r^* - n < n < B$. Thus, $r_1 = r^* - n$. If $(r_1 > q_0)$ is false, the final conditional `if` $(r_1 < n)$ acts and correctly returns $r_1 = r^* - n$. If $(r_1 > q_0)$ is true, the algorithm adds n, so $r_1 \leftarrow r^*$. The final conditional sees $r^* \geq n$, and the `else` branch returns $r^* - n$. Both paths perfectly return the correct remainder $r^* - n$.

Case 2: $2n \leq r^* < 3n$ (Quotient q_1 is off by 2). Here, $n \leq r^* - n < 1.25B - B/2 = 0.75B < B$. Thus, $r_1 = r^* - n$. To output the correct remainder $r^* - 2n$, the algorithm **must not** add n. Therefore, $(r_1 > q_0)$ must be strictly FALSE. We prove this by contradiction. Assume $(r_1 > q_0)$ is TRUE.

$$r^* - n > q_0 \implies \frac{u_0 + q_0}{2} + \gamma k - B/2 - k > q_0 \implies u_0 - q_0 + 2\gamma k > B + 2k$$

Additionally, the condition $r^* \geq 2n$ implies:

$$\frac{u_0 + q_0}{2} + \gamma k \geq B + 2k \implies q_0 + u_0 + 2\gamma k \geq 2B + 4k$$

Summing these two inequalities cancels q_0:

$$2u_0 + 4\gamma k > 3B + 6k$$

Because $u_0 \leq B - 1$, we know $2u_0 < 2B$. Substituting this yields:

$$2B + 4\gamma k > 3B + 6k \implies 4\gamma k > B + 6k$$

Since $\gamma \leq 4k$, the left side is at most $16k^2$. Therefore:

$$16k^2 > B + 6k$$

However, the initial conjecture constraint definitively states that $16k^2 \leq B$. It is mathematically impossible for $B \geq 16k^2 > B + 6k$ to be true. This contradiction proves that $(r_1 > q_0)$ must be FALSE. The algorithm correctly avoids adding n, and the final `else` branch returns $r_1 - n = r^* - 2n$. □