Kyber Implementation Notes

Goutam Tamvada

January 23, 2024

Contents

1 Preliminaries

We assume the natural numbers $\mathbf{N}$ start at 0. Let $\ggg$ denote an arithmetic right-shift, and & the bitwise AND operation. We use the exclamation mark '!' to denote the bitwise NOT operator.

1.1 Floors and ceilings

We make use of the following lemmas, stated here without proof:

Lemma 1. *If n is an integer and x is a real number, $\lfloor n + x \rfloor = n + \lfloor x \rfloor$.*

Lemma 2. *If $x \in \mathbf{R}$ is not an integer, then:*

$$\lceil x \rceil = \lfloor x \rfloor + 1$$

Lemma 3. *If $x \in \mathbf{R}$ and $m, n \in \mathbf{Z}$:*

$$\left\lfloor \frac{m + x}{n} \right\rfloor = \left\lfloor \frac{m + \lfloor x \rfloor}{n} \right\rfloor$$

By $\lfloor x \rceil$ we mean $\lfloor x + \frac{1}{2} \rfloor$.

1.2 Signed and unsigned representatives

For $N \in \mathbf{N}$, let

$$U_N = \{0, 1, \ldots, N-1\}$$
$$S_N = \left\{ -\left\lfloor \frac{N}{2} \right\rfloor, -\left\lfloor \frac{N}{2} \right\rfloor + 1, \ldots, \left\lfloor \frac{N-1}{2} \right\rfloor - 1, \left\lfloor \frac{N-1}{2} \right\rfloor \right\}$$

Some facts about S_N:

Lemma 4. *$\lfloor \frac{N-1}{2} \rfloor$ is the greatest integer less than $\frac{N}{2}$.*

Proof.

Case 1: *N is odd*

Then $N = 2n + 1$ for some $n \in \mathbf{N}$ and:

$$\begin{aligned} \left\lfloor \frac{N-1}{2} \right\rfloor &= \left\lfloor \frac{2n+1-1}{2} \right\rfloor \\ &= \lfloor n \rfloor \\ &= n \end{aligned}$$

This is clearly less than $n + \frac{1}{2} = \frac{N}{2}$; there is furthermore no integer k satisfying $n < k < n + \frac{1}{2}$.

Case 2: *N is even*

Then $N = 2n$ for some $n \in \mathbf{N}$ and:

$$\begin{aligned}\left\lfloor \frac{N-1}{2} \right\rfloor &= \left\lfloor \frac{2n-1}{2} \right\rfloor \\ &= \left\lfloor n - \frac{1}{2} \right\rfloor \\ &= n - 1\end{aligned}$$

This is clearly less than $n = \frac{N}{2}$; there is furthermore no integer k satisfying $n - 1 < k < n$. □

Lemma 5. *$-\left\lfloor \frac{N}{2} \right\rfloor$ is the smallest integer that is greater than or equal to $-\frac{N}{2}$.*

Proof. If N is even then $-\left\lfloor \frac{N}{2} \right\rfloor = -\frac{N}{2}$. If N is odd then it has the form $2n+1$ for some natural number n, and as such, $-\left\lfloor \frac{N}{2} \right\rfloor = -\left\lfloor n + \frac{1}{2} \right\rfloor = -n > -n - \frac{1}{2} = -\frac{N}{2}$. Furthermore, there is no integer k such that $-n > k > -n - \frac{1}{2}$. □

Lemma 6. *S_N has N elements.*

Proof.

Case 1: *N is odd*

Then $N = 2n + 1$ for some $n \in \mathbf{N}$ and because:

$$\begin{aligned}-\left\lfloor \frac{N}{2} \right\rfloor &= -\left\lfloor \frac{2n+1}{2} \right\rfloor \\ &= -\left\lfloor n + \frac{1}{2} \right\rfloor \\ &= -n\end{aligned}$$

and

$$\begin{aligned}\left\lfloor \frac{N-1}{2} \right\rfloor &= \left\lfloor \frac{2n+1-1}{2} \right\rfloor \\ &= \lfloor n \rfloor \\ &= n\end{aligned}$$

S_N has $\left\lfloor \frac{N-1}{2} \right\rfloor - (-\left\lfloor \frac{N}{2} \right\rfloor) + 1 = n - (-n) + 1 = 2n + 1 = N$ elements.

Case 2: *N is even*

Then $N = 2n$ for some $n \in \mathbf{N}$ and because:

$$\begin{aligned} -\left\lfloor \frac{N}{2} \right\rfloor &= -\left\lfloor \frac{2n}{2} \right\rfloor \\ &= -n \end{aligned}$$

and

$$\begin{aligned} \left\lfloor \frac{N-1}{2} \right\rfloor &= \left\lfloor \frac{2n-1}{2} \right\rfloor \\ &= \left\lfloor n - \frac{1}{2} \right\rfloor \\ &= n-1 \end{aligned}$$

S_N has $\left\lfloor \frac{N-1}{2} \right\rfloor - (-\left\lfloor \frac{N}{2} \right\rfloor) + 1 = (n-1) - (-n) + 1 = 2n = N$ elements. □

For $v \in \mathbf{Z}$, let $v \bmod N = v - N\left\lfloor \frac{v}{N} \right\rfloor$. It is clear that $v \bmod N$ maps v to a unique value in U_N; we call this value the *canonical unsigned representative* of v.

Now let $v \bmod {}^{\pm}N = v - N\left\lfloor \frac{v}{N} + \frac{1}{2} \right\rfloor$. Due to the following theorems, we call $v \bmod {}^{\pm}N$ the *canonical signed representative* of v.

Theorem 1. *$v \bmod {}^{\pm}N$ maps v to a unique value in S_N.*

Proof. There must be a unique $k \in \mathbf{Z}$ such that $kN \leq v < (k+1)N$. v could lie in one of two intervals:

Case 1: $kN \leq v < (k + \frac{1}{2})N$

This implies that $k + \frac{1}{2} \leq \frac{v}{N} + \frac{1}{2} < k+1$ thereby implying that $\left\lfloor \frac{v}{N} + \frac{1}{2} \right\rfloor = k$. This in turn means that $v \bmod {}^{\pm}N = v - Nk$ and so we have: $0 \leq v \bmod {}^{\pm}N < \frac{N}{2}$.

Case 2: $(k + \frac{1}{2})N \leq v < (k+1)N$

This implies that $k+1 \leq \frac{v}{N} + \frac{1}{2} < k + \frac{3}{2}$ thereby implying that $\left\lfloor \frac{v}{N} + \frac{1}{2} \right\rfloor = k+1$. This in turn means that $v \bmod {}^{\pm}N = v - N(k+1)$ and so we have: $-\frac{N}{2} \leq v \bmod {}^{\pm}N < 0$.

From this and from Lemmas 4, 5, 6, the statement of the theorem follows. □

Theorem 2. *$a \bmod {}^{\pm}N = b \bmod {}^{\pm}N \iff a \equiv b \pmod N$.*

Proof. If $a \equiv b \pmod{N}$ then $a = b + Ny$ for some $y \in \mathbf{Z}$, and so:

$$\begin{aligned} a \bmod^{\pm} N = (b + Ny) \bmod^{\pm} N &= (b + Ny) - N \left\lfloor \frac{(b + Ny)}{N} + \frac{1}{2} \right\rfloor \\ &= b + Ny - N \left\lfloor \frac{b}{N} + \frac{1}{2} + y \right\rfloor \\ &= b + Ny - N \left\lfloor \frac{b}{N} + \frac{1}{2} \right\rfloor - Ny \\ &= b - N \left\lfloor \frac{b}{N} + \frac{1}{2} \right\rfloor \\ &= b \bmod^{\pm} N \end{aligned}$$

On the other hand, if $a \bmod^{\pm} N = b \bmod^{\pm} N$ then:

$$\begin{aligned} a - N \left\lfloor \frac{a}{N} + \frac{1}{2} \right\rfloor &= b - N \left\lfloor \frac{b}{N} + \frac{1}{2} \right\rfloor \\ \implies a - b &= N \left(\left\lfloor \frac{a}{N} + \frac{1}{2} \right\rfloor - \left\lfloor \frac{b}{N} + \frac{1}{2} \right\rfloor \right) \end{aligned}$$

Since $\left\lfloor \frac{a}{N} + \frac{1}{2} \right\rfloor$ and $\left\lfloor \frac{b}{N} + \frac{1}{2} \right\rfloor$ are integers, $a \equiv b \pmod{N}$. □

Lemma 7. *The maximum value of* $|v \bmod^{\pm} N|$ *is* $\frac{N}{2}$.

Proof. This is evident from Lemmas 4, 5, 6 and from Theorem 1. □

1.3 Bit operations and bit representations

Lemma 8. *Let* $R = 2^b$ *for some* $b \in \mathbf{N}$*, let* $v \geq R$*. Then,* $v \bmod^{\pm} R$ *can be computed by interpreting the result of* $v \;\&\; (R - 1)$ *as being in two's-complement representation.*

Proof. Let v be a k-bit value. $k - 1 \geq b$ by assumption, and so let $v = v_{k-1}2^{k-1} + v_{k-2}2^{k-2} + v_{b+1}2^{b+1} + v_b 2^b + v_{b-1}2^{b-1} \cdots + v_0$ be the binary expansion of v. With this, we have:

$$\begin{aligned} R \left\lfloor \frac{v}{R} + \frac{1}{2} \right\rfloor &= 2^b \left\lfloor v_{k-1}2^{k-1-b} + v_{k-2}2^{k-2-b} \cdots + v_b + \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} \cdots + \frac{v_0}{2^b} + \frac{1}{2} \right\rfloor \\ &= v_{k-1}2^{k-1} + v_{k-2}2^{k-2} \cdots + v_b 2^b + 2^b \left\lfloor \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} \cdots + \frac{v_0}{2^b} + \frac{1}{2} \right\rfloor \end{aligned}$$

This means that $v - R \left\lfloor \frac{v}{R} + \frac{1}{2} \right\rfloor$ is:

$$v_{b-1}2^{b-1} + \cdots + v_0 - 2^b \left\lfloor \frac{v_{b-1}}{2} + \cdots + \frac{v_0}{2^b} + \frac{1}{2} \right\rfloor$$

Case 1: v_{b-1} *is 0*

This means that $0 \leq v_{b-1}2^{b-1} + v_{b-2}2^{b-2} + \cdots + v_0 < 2^{b-1}$ (in fact it is at most $2^{b-1} - 1$). Dividing by 2^b, we see that:

$$\begin{aligned} 0 \leq \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + v_0 < \frac{1}{2} \\ \implies 0 \leq \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + v_0 + \frac{1}{2} < 1 \end{aligned}$$

And so we get $v - R\left\lfloor \frac{v}{R} + \frac{1}{2} \right\rfloor = v_{b-2}2^{b-2} + v_{b-3}2^{b-3} + \cdots + v_0$. Notice that this is equivalent to the two's-complement signed interpretation of v & $(R-1)$ where the "sign bit" v_{b-1} is 0.

Case 2: v_{b-1} *is 1*

This means that $2^{b-1} \leq v_{b-1}2^{b-1} + v_{b-2}2^{b-2} + \cdots + v_0 < 2^b$. Dividing by 2^b, we see that:

$$\begin{aligned} \frac{1}{2} \leq \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + v_0 < 1 \\ \implies 1 \leq \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + v_0 + \frac{1}{2} < \frac{3}{2} \end{aligned}$$

And so we get

$$\begin{aligned} v - R\left\lfloor \frac{v}{R} + \frac{1}{2} \right\rfloor &= v_{b-1}2^{b-1} + v_{b-2}2^{b-2} + \cdots + v_0 - 2^b \\ &= (-1)2^b + (1)2^{b-1} + v_{b-2}2^{b-2} + \cdots + v_0 \\ &= -2^{b-1}(2-1) + v_{b-2}2^{b-2} + \cdots + v_0 \\ &= -(1)2^{b-1} + v_{b-2}2^{b-2} + \cdots + v_0 \end{aligned}$$

Notice that this is equivalent to the two's-complement signed interpretation of v & $(R-1)$ where the "sign bit" v_{b-1} is 1.

□

Lemma 9. *Let v be some k-bit value representing a number using two's-complement representation, and let $0 \leq b < k$. Then, $v \ggg k = \left\lfloor \frac{v}{2^b} \right\rfloor$.*

Proof. Let $i = k - b$ so that v can be written as $-v_{b+(i-1)}2^{b+(i-1)} + v_{b+(i-2)}2^{b+(i-2)} + \cdots + v_{b+1}2^{b+1} + v_b 2^b + v_{b-1}2^{b-1} + \cdots + v_0$.

Case 1: $v_{b+(i-1)}$ *is 1*

Observe that:

$$\begin{aligned} v \ggg k &= (-1)2^{b+(i-1)} + (1)2^{b+(i-2)} + \cdots + (1)2^{b+(i-(b+1))} + v_{b+(i-2)}2^{i-2} + \cdots + v_b \\ &= 2^{b+(i-1)}(-1 + \frac{1}{2} + \cdots + \frac{1}{2^b}) + v_{b+(i-2)}2^{i-2} + \cdots + v_b \\ &= 2^{b+(i-1)}(-1 + (1/2)\frac{1-(1/2)^b}{1/2}) + v_{b+(i-2)}2^{i-2} + \cdots + v_b \\ &= -\frac{2^{b+(i-1)}}{2^b} + v_{b+(i-2)}2^{i-2} + v_{b+(i-3)}2^{i-3} + \cdots + v_b \end{aligned}$$

and that:

$$\begin{aligned} \left\lfloor \frac{v}{2^b} \right\rfloor &= \left\lfloor -\frac{2^{b+(i-1)}}{2^b} + v_{b+(i-2)}2^{i-2} + v_{b+(i-3)}2^{i-3} + \cdots + v_b + \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + \frac{v_0}{2^b} \right\rfloor \\ &= -\frac{2^{b+(i-1)}}{2^b} + v_{b+(i-2)}2^{i-2} + v_{b+(i-3)}2^{i-3} + \cdots + v_b + \left\lfloor \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + \frac{v_0}{2^b} \right\rfloor \end{aligned}$$

Now, the maximum value that $\frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + \frac{v_0}{2^b}$ can take is $\frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{2^b} = (1/2)\frac{1-(1/2)^b}{1-1/2} = 1 - \frac{1}{2^b}$. This value is lesser than 1, which means $\lfloor \frac{v_{b-1}}{2} + \frac{v_{b-2}}{4} + \cdots + \frac{v_0}{2^b} \rfloor = 0$, and so the statement of the lemma follows.

Case 2: $v_{b+(i-1)}$ *is 0*

With the note that in this case, an arithmetic right shift is equivalent to a logical one, the proof proceeds in a manner similar to above.

□

2 Signed Montgomery Reduction

Suppose we want to reduce some $v \in \mathbf{Z}$ with respect to an odd modulus N. Let $R = 2^b$ for some $b \in \mathbf{N}$. Since N is odd and R is even, both $N^{-1} \bmod R$ and $R^{-1} \bmod N$ exist.

- If v is a multiple of R, then $\frac{v}{R} \equiv vR^{-1} \pmod N$.
- If v is not a multiple of R, we *correct* it by finding a k such that $v - kN$ is divisible by R, thereby giving us a representative $\frac{v-kN}{R} \equiv vR^{-1} \pmod N$. Such a k would have to satisfy the congruence $v \equiv kN \pmod R$. In other words $k \equiv vN^{-1} \pmod R$. The smallest such k is obtained by setting $k = vN^{-1} \bmod {}^{\pm}R$.

These insights lead us to:

Algorithm 1 Signed Montgomery Reduction

Input: odd modulus N
Input: $R = 2^b$ for some $b \in \mathbf{N}$.
Input: integer v to be reduced
Output: integer o such that $o \equiv vR^{-1} \pmod{N}$
1: $T \leftarrow N^{-1} \bmod^{\pm} R$
2: $k \leftarrow vT \bmod^{\pm} R$
3: $c \leftarrow \left\lfloor \frac{kN}{R} \right\rfloor$
4: $v_{\text{high}} \leftarrow \left\lfloor \frac{v}{R} \right\rfloor$
5: **return** $o \leftarrow v_{\text{high}} - c$

Note that if N and R are taken to be fixed parameters, the quantity T in step 1 can be input to the algorithm as a pre-computed constant.

To prove the correctness of this algorithm, we also need:

Lemma 10. *If it holds for some integers R, a, b that R divides $(a-b)$, then $\frac{a-b}{R} = \left\lfloor \frac{a}{R} \right\rfloor - \left\lfloor \frac{b}{R} \right\rfloor$.*

Proof. Since R divides $a - b$, $\frac{a-b}{R}$ = some integer k. This means that $\frac{a}{R} = k + \frac{b}{R}$, which in turn means that $\left\lfloor \frac{a}{R} \right\rfloor = k + \left\lfloor \frac{b}{R} \right\rfloor$. Thus:

$$\begin{aligned} \left\lfloor \frac{a}{R} \right\rfloor - \left\lfloor \frac{b}{R} \right\rfloor &= k + \left\lfloor \frac{b}{R} \right\rfloor - \left\lfloor \frac{b}{R} \right\rfloor \\ &= k \\ &= \frac{a-b}{R} \end{aligned}$$

□

We now can prove:

Theorem 3. *Algorithm 1 is correct.*

Proof. Due to our choice of $k = vT \bmod^{\pm} R$ we have: $k \equiv vT \pmod{R} \implies kN \equiv v \pmod{R}$, which implies R divides $v - kN$. Using Lemma 10 with $a = v$ and $b = kN$, we see that $o = \frac{v-kN}{R}$. And so we have:

$$
\begin{aligned}
& v - kN \equiv v \pmod{N} \\
\implies & v - kN \equiv vRR^{-1} \pmod{N} \\
\implies & \frac{v - kN}{R} \equiv vR^{-1} \pmod{N} \\
\implies & o \equiv vR^{-1} \pmod{N}
\end{aligned}
$$

□

We went through all this trouble so that we could make the output representative smaller as follows:

Theorem 4. *The output of Algorithm 1 satisfies* $|o| \leq \frac{|v|}{R} + \frac{N}{2}$. *In particular, if* $|v| \leq \frac{NR}{2}$ *then* $|o| \leq N$ *and if* $|v| \leq NR$ *then* $|o| \leq \frac{3N}{2}$.

Proof. We have $|o| = |\frac{v-kN}{R}| \leq |\frac{v}{R}| + |\frac{kN}{R}|$.

Since $k = vT \bmod^{\pm} R$, by Lemma 7 we see that $|k|$ can take a maximum value of $|\frac{R}{2}|$, which implies $|\frac{kN}{R}| \leq \frac{N}{2}$. □

2.1 Signed Montgomery reduction in `libcrux`'s implementation of Kyber

`libcrux`'s implementation of Montgomery reduction (for the Kyber KEM) uses $N = 3329$ and $R = 2^{16}$. The implementation (in Rust) is reproduced below (`KyberFieldElement` is an alias for `i32`):

```
1  const MONTGOMERY_SHIFT: i32 = 16;
2  const MONTGOMERY_R: i32 = 1i32 << MONTGOMERY_SHIFT;
3  const INVERSE_OF_MODULUS_MOD_R: i32 = -3327; // FIELD_MODULUS^{-1} mod
   ↪ MONTGOMERY_R
4
5  pub(crate) fn montgomery_reduce(value: KyberFieldElement) ->
   ↪ KyberFieldElement {
6      let k = (value & (MONTGOMERY_R - 1)) * INVERSE_OF_MODULUS_MOD_R;
7      let k = (k & (MONTGOMERY_R - 1)) as i16;
8
9      let c = (i32::from(k) * FIELD_MODULUS) >> MONTGOMERY_SHIFT;
10     let value_high = value >> MONTGOMERY_SHIFT;
11
12     value_high - c
13 }
```

Let us go through this implementation line by line and prove that it implements Algorithm 1. For ease of exposition we will refer to `value` as v.

```
let k = (value & (MONTGOMERY_R - 1)) * INVERSE_OF_MODULUS_MOD_R;
let k = (k & (MONTGOMERY_R - 1)) as i16;
```

Since $R = 2^{16}$ and v is a 32-bit (signed) quantity, v can be written as $v_1 R + v_0$, where v_0 and v_1 are 16-bit quantities. Now, by Lemma 8, these two lines of code compute $k = v_0 T \bmod^{\pm} R$. We can get away with this since:

$$k \equiv v_0 T \equiv v_0 T + v_1 RT \equiv (v_0 + v_1 R)T \equiv vT \pmod{R}$$

```
let c = (i32::from(k) * FIELD_MODULUS) >> MONTGOMERY_SHIFT;
let value_high = value >> MONTGOMERY_SHIFT;

value_high - c
```

By Lemma 9, we see that $c = \left\lfloor \frac{kN}{R} \right\rfloor$ and that $v_{\text{high}} = \left\lfloor \frac{v}{R} \right\rfloor$.

3 Signed Barrett Reduction

. Letting R be some power of 2, since:

$$v \bmod^{\pm} N = v - N \left\lfloor \frac{v}{N} \right\rceil = v - N \left\lfloor \frac{v}{R} \cdot \frac{R}{N} \right\rceil$$

we can approximate $\frac{R}{N}$ to some nearby integer, and, if A denotes this approximation, we get

$$v \bmod^{\pm} N \approx v - N \left\lfloor \frac{vA}{R} \right\rceil$$

As we will soon see, taking A to be $\left\lfloor \frac{R}{N} \right\rceil$ (as opposed to $\left\lfloor \frac{R}{N} \right\rfloor$ or $\left\lceil \frac{R}{N} \right\rceil$) produces the smallest representative in absolute value. Without further ado:

Algorithm 2 Signed Barrett Reduction

Input: odd modulus N
Input: $R = 2^b$ for some $b \in \mathbf{N}$
Input: integer v to be reduced
Output: integer o such that $o \equiv v \pmod{N}$

1: $A \leftarrow \left\lfloor \frac{R}{N} \right\rceil$
2: $k \leftarrow \left\lfloor \frac{vA}{R} \right\rceil$
3: $c \leftarrow Nk$
4: **return** $o \leftarrow v - c$

Note that if N and R are taken to be fixed parameters, the quantity A in step 1 can be input to the algorithm as a pre-computed constant.

Theorem 5. *Algorithm 2 is correct.*

Proof. In simply expanding o we get:

$$o = v - c = v - Nt \equiv v \pmod{N}$$

□

More interesting is the magnitude of the output representative o. First:

Lemma 11. $A = \frac{R - R \bmod^{\pm} N}{N}$

Proof. This is merely a rearrangement of the definition of $R \bmod^{\pm} N = R - N \left\lfloor \frac{R}{N} \right\rceil = R - N \cdot A$. □

Second, since N is odd and R is even, $N^{-1} \bmod^{\pm} R$ exists; let $T = N^{-1} \bmod^{\pm} R$.

Lemma 12. $vA \bmod^{\pm} R = (-v(R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R$

Proof. Since from Lemma 11 we have $N \cdot A = R - R \bmod^{\pm} N$:

$$\begin{aligned} N \cdot A &\equiv (-R \bmod^{\pm} N) \pmod{R} \\ \implies NA \cdot T &\equiv -(R \bmod^{\pm} N) \cdot T \pmod{R} \\ \implies A &\equiv -(R \bmod^{\pm} N) \cdot T \pmod{R} \\ \implies vA &\equiv -v(R \bmod^{\pm} N) \cdot T \pmod{R} \end{aligned}$$

The statement of the lemma follows from applying Theorem 2 to this last expression. □

Theorem 6. *The output o of Algorithm 2 satisfies $|o| \leq \frac{|v| \cdot \frac{N}{2}}{R} + \frac{N}{2}$. In particular, if $|v| < R$ then $|o| < N$.*

Proof. We have:

$$o = v - c = v - Nk = v - N \left\lfloor \frac{vA}{R} \right\rceil$$

Since $vA \bmod^{\pm} R = vA - R \left\lfloor \frac{vA}{R} \right\rceil$, we have $\left\lfloor \frac{vA}{R} \right\rceil = \frac{vA - vA \bmod^{\pm} R}{R}$. The expression for o therefore becomes:

$$o = v - N \left\lfloor \frac{vA}{R} \right\rceil \tag{1}$$

$$= v - N \left(\frac{vA - vA \bmod^{\pm} R}{R} \right) \tag{2}$$

$$= v - N \left(\frac{v(\frac{R - R \bmod^{\pm} N}{N}) - vA \bmod^{\pm} R}{R} \right) \text{ (using Lemma 11)} \tag{3}$$

$$= v - N \left(\frac{v(\frac{R - R \bmod^{\pm} N}{N}) - [(v \cdot (R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R]}{R} \right) \text{ (using Lemma 12)} \tag{4}$$

$$= v - \frac{v(R - R \bmod^{\pm} N) - N[(v \cdot (R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R]}{R} \text{ (moving } N \text{ into the bracket)} \tag{5}$$

$$= v + \frac{-vR + v(R \bmod^{\pm} N) + N[(v \cdot (R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R]}{R} \tag{6}$$

$$= \frac{v(R \bmod^{\pm} N) + N[(v(R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R]}{R} \tag{7}$$

$$= \frac{v(R \bmod^{\pm} N)}{R} + \frac{N[(v \cdot N \cdot (R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R]}{R} \tag{8}$$

Since by Lemma 7 the maximum value of $|R \bmod^{\pm} N|$ is $\frac{N}{2}$ and the maximum value of $|(v \cdot N \cdot (R \bmod^{\pm} N) \cdot T) \bmod^{\pm} R|$ is $\frac{R}{2}$, we get:

$$|o| \leq \frac{|v| \cdot \frac{N}{2}}{R} + \frac{N \cdot \frac{R}{2}}{R}$$

from which the statement of the theorem follows.

□

Suppose instead we took A to be $\left\lfloor \frac{R}{N} \right\rfloor$, we would then have to substitute $\frac{R - R \bmod N}{N}$ into 3 above and get $\frac{v(R \bmod N)}{R}$ instead of $\frac{v(R \bmod^{\pm} N)}{R}$ in 8. This would result in a larger output representative since the maximum absolute value of $R \bmod N$ is $N - 1$.

3.1 Signed Barrett reduction in `libcrux`'s implementation of Kyber

`libcrux`'s implementation of Barrett reduction (for the Kyber KEM) uses $N = 3329$ and $R = 2^{26}$. The implementation (in Rust) is reproduced below with attributes and `debug_asserts` removed (`KyberFieldElement` is an alias for `i32`).

```
const BARRETT_SHIFT: i64 = 26;
const BARRETT_R: i64 = 1 << BARRETT_SHIFT;
const BARRETT_MULTIPLIER: i64 = 20159; // floor((BARRETT_R / FIELD_MODULUS)
↪  + 0.5)

pub(crate) fn barrett_reduce(value: KyberFieldElement) -> KyberFieldElement
↪  {
    let t = (i64::from(value) * BARRETT_MULTIPLIER) + (BARRETT_R >> 1);
    let quotient = (t >> BARRETT_SHIFT) as i32;

    let result = value - (quotient * FIELD_MODULUS);

    result
}
```

Once again, let us go through this implementation line by line and prove that it implements Algorithm 2. For ease of exposition we will refer to `value` as v.

```
    let t = (i64::from(value) * BARRETT_MULTIPLIER) + (BARRETT_R >> 1);
    let quotient = (t >> BARRETT_SHIFT) as i32;
```

Notice that `BARRETT_MULTIPLIER` is the precomputed $A = \left\lfloor \frac{R}{N} \right\rceil$ and that:

$$\begin{aligned}
\left\lfloor \frac{vA}{R} \right\rceil &= \left\lfloor \frac{vA}{R} + \frac{1}{2} \right\rfloor \\
&= \left\lfloor \frac{2vA + R}{2R} \right\rfloor \\
&= \left\lfloor \frac{2\left(vA + \frac{R}{2}\right)}{2R} \right\rfloor \\
&= \left\lfloor \frac{vA + \frac{R}{2}}{R} \right\rfloor
\end{aligned}$$

Line 1 computes $t = vA + \frac{R}{2}$ (as an `i64`) and line 2 code computes $\left\lfloor \frac{t}{R} \right\rfloor$. Furthermore, since $R = 2^{26}$ and $\lceil log_2(A) \rceil = \lceil log_2(20159) \rceil = 15$, if $|v| < R$:

$$\left| vA + \frac{R}{2} \right| < 2^{26} \cdot 2^{15} + 2^{25} = 2^{25} \cdot (2^{16} + 1) \tag{9}$$

$$\implies \left| \frac{vA + \frac{R}{2}}{R} \right| < 2^{15} + \frac{1}{2} \tag{10}$$

We can be certain from 9 that `t` will not overflow and from 10 that `quotient` will not overflow.

The remaining code can easily be seen to match the specification of Algorithm 2.

4 Serialization

Once we're all done with computation, we need to serialize the resulting representatives. In the Kyber KEM specification, all serialization is done on canonical unsigned representatives. For representatives in S_N, we can simply add N to the ones less than 0 and get the corresponding canonical unsigned representative. This input range can be slightly widened as follows:

For some modulus N, define $\texttt{shift} : (-N, N) \to U_N$:

$$\texttt{shift}(z) = \begin{cases} z + N & \text{if } z < 0 \\ z & \text{otherwise} \end{cases}$$

We then have:

Theorem 7. *For some $N \in \mathbf{N}$, suppose we have two representatives $r_1 \in S_N$ and $r_2 \in (-N, N)$ such that $r_1 \equiv r_2 \pmod{N}$. Then we have: $\texttt{shift}(r_1) = \texttt{shift}(r_2)$*

Proof.

Case 1: $0 \leq r_1$ *and* $0 \leq r_2$.

Since r_1 and r_2 are congruent mod N, $r_1 \bmod N = r_2 \bmod N$. Since $0 \leq r_1 < \left\lfloor \frac{N-1}{2} \right\rfloor$ and $0 \leq r_2 < N$, this means $r_1 = r_2$.

Case 2: $0 \leq r_1$ *and* $r_2 < 0$.

Since $r_1 \equiv r_2 \pmod{N}$, $r_1 \equiv r_2 + N \pmod{N}$. Since $-\frac{N}{2} \leq r_2 < 0$ we have $\frac{N}{2} \leq r_2 + N < N$. This means $r_1 \bmod N = r_2 + N = (r_2 + N) \bmod N$. Since r_1 is non-negative and r_2 negative:

$$\texttt{shift}(r_1) = r_1 = r_2 + N = \texttt{shift}(r_2)$$

Case 3: $r_1 < 0$ *and* $0 \leq r_2$.

Since $r_1 \equiv r_2 \pmod{N}$, $r_1 + N \equiv r_2 \pmod{N}$. Since $-\frac{N}{2} \leq r_1 < 0$ we have $\frac{N}{2} \leq r_1 + N < N$. This means $(r_1 + N) \bmod N = r_1 + N = r_2 = r_2 \bmod N$. Since r_1 is negative and r_2 non-negative:

$$\texttt{shift}(r_1) = r_1 + N = r_2 = \texttt{shift}(r_2)$$

Case 4: $r_1 < 0$ *and* $r_2 < 0$.

Since $r_1 \equiv r_2 \pmod{N}$, $r_1 + N \equiv r_2 + N \pmod{N}$. Since $-\frac{N}{2} \leq r_1 < 0$ we have $\frac{N}{2} \leq r_1 + N < N$; the same holds for r_2. This means $(r_1 + N) \bmod N = r_1 + N = r_2 + N = (r_2 + N) \bmod N$. Since both r_1 and r_2 are negative:

$$\texttt{shift}(r_1) = r_1 + N = r_2 + N = \texttt{shift}(r_2)$$

□

`shift` is exactly the function `to_unsigned_representative` used in the `libcrux` Rust implementation of the Kyber KEM, and, as it happens, $(-N, N)$ is the output range guaranteed by the signed Barrett reduction implementation in `libcrux`.

5 Constant-time operations

Suppose we have two Rust `u16` variables `a` and `b`. The operation `a.wrapping_add(b)` is defined as $(a + b) \bmod 2^{16}$. The following theorem provides a method to determine, using constant-time operations, whether a value $a \in [0, 2^8)$ is zero or non-zero.

Theorem 8. *Suppose we have a* `u16` `a` *such that* $0 \leq \texttt{a} < 2^8$. *Let* `r : u16 = (!a).wrapping_add(1)`. *If* `a` *is 0, then* `r` *is 0. Otherwise, bits 9-16 of* `r` *(with bit 0 being the least significant bit) will always be set to 1.*

Proof. If `a` is 0, `(!a)` is $2^{16} - 1$, and so `(!a).wrapping_add(1)` $= 2^{16} - 1 + 1 \bmod 2^{16} = 0$.

Now suppose `a` is not 0. Since it is less than 2^8, its binary expansion can be written as:

$$(0)2^{15} + (0)2^{14} + \cdots + (0)2^8 + a_7 2^7 + a_6 2^6 + \cdots + a_0$$

and therefore the bitwise NOT of `a` can be written as:

$$\begin{aligned}
\texttt{!a} &= (1-0)2^{15} + (1-0)2^{14} + \cdots + (1-0)2^8 + (1-a_7)2^7 + (1-a_6)2^6 + \cdots + (1-a_0) \\
&= (2^{15} + 2^{14} + \cdots + 2^8) + (2^7 + 2^6 + \cdots + 1) - (a_7 2^7 + a_6 2^6 + \cdots + a_0) \\
&= 2^{16} - 1 - \texttt{a}
\end{aligned}$$

Which in turn implies that

$$\begin{aligned}
\texttt{(!a).wrapping_add(1)} &= (2^{16} - 1 - \texttt{a} + 1) \bmod 2^{16} \\
&= (2^{16} - \texttt{a}) \bmod 2^{16} \\
&= (2^{16} - \texttt{a}) \;\; (0 < 2^{16} - \texttt{a} < 2^{16} \text{ since } 0 < \texttt{a} < 2^8)
\end{aligned}$$

Now the i^{th} bit of $2^{16} - \texttt{a}$ is:

$$\left\lfloor \frac{2^{16} - \texttt{a}}{2^{i-1}} \right\rfloor \bmod 2 = \left\lfloor \frac{2^{16}}{2^{i-1}} - \frac{a}{2^{i-1}} \right\rfloor \bmod 2$$

For bits 9-16 we have $8 \leq i - 1 \leq 15 \implies \frac{2^{16}}{2^{i-1}} = 2^{16-(i-1)}$ is always an even number. We also have $\frac{\texttt{a}}{2^8} \geq \frac{\texttt{a}}{2^{i-1}} \geq \frac{\texttt{a}}{2^{15}}$. Since $0 < \texttt{a} < 2^8$ this means

$$\begin{aligned}
0 < \frac{\texttt{a}}{2^{15}} < \frac{\texttt{a}}{2^{i-1}} < \frac{\texttt{a}}{2^8} < 1 \\
\implies 0 > -\frac{\texttt{a}}{2^{i-1}} > -1 \\
\implies \left\lfloor -\frac{\texttt{a}}{2^{i-1}} \right\rfloor = -1
\end{aligned}$$

From all of this we get:

$$\left\lfloor \frac{2^{16}}{2^{i-1}} - \frac{a}{2^{i-1}} \right\rfloor \bmod 2 = 2^{16-(i-1)} - 1 \bmod 2$$

$2^{16-(i-1)} - 1$ is always odd, which means $2^{16-(i-1)} - 1 \bmod 2$ is always 1.

□

Theorem 9. *If negative integers are represented in 16 bits using two's-complement signed representation,* $\texttt{shift}(z) = z + ((z \ggg 15)\ \&\ N)$.

6 Flooring Divisions Without Dividing

Let Q be some fixed constant that is not a power of 2. We want to calculate the quantity $\lfloor \frac{n}{Q} \rfloor$ for some $n \in \mathbf{N}$ without actually performing any divisions.

We can borrow the trick used in Barrett Reduction to do so: for some positive natural number k we have:

$$\left\lfloor \frac{n}{Q} \right\rfloor = \left\lfloor \frac{n}{2^k} \times \frac{2^k}{Q} \right\rfloor$$

If we could use some constant C in place of the value $\frac{2^k}{Q}$, then to compute $\lfloor \frac{n}{Q} \rfloor$ we would only need to multiply n by C and right-shift the product, viewed as a binary number, by k bits.

So what should k and C be? First we need two lemmas.

Lemma 13. *The fractional part of $\frac{n}{Q}$ is at most $\frac{Q-1}{Q}$.*

Proof. n can be expressed as $Qb+r$ for some quotient b and some remainder r such that $0 \leq r < Q$. This means that $\frac{n}{Q} = b + \frac{r}{Q}$. The statement of the lemma follows from observing that b is an integer, and r can be at most $Q - 1$. □

Lemma 14. *Consider a non-negative integer z that is not divisible by Q. Then:*

$$\left\lceil \frac{z}{Q} \right\rceil = \frac{z + f}{Q}$$

where $f = Q - z \bmod Q$ and $0 < f < Q$.

(Intuitively, f represents how "far" z is from the next multiple to Q)

Proof. Observe that:

$$\begin{aligned} z \bmod Q &= z - Q \left\lfloor \frac{z}{Q} \right\rfloor \\ \implies z - z \bmod Q &= Q \left\lfloor \frac{z}{Q} \right\rfloor \\ \implies \frac{z - z \bmod Q}{Q} &= \left\lfloor \frac{z}{Q} \right\rfloor \end{aligned}$$

z not being divisible by Q implies that $\frac{z}{Q}$ is not an integer, which means:

$$\begin{aligned}\frac{z+f}{Q} &= \frac{z+Q-z \bmod Q}{Q} \\ &= \frac{z-z \bmod Q}{Q} + \frac{Q}{Q} \\ &= \left\lfloor \frac{z}{Q} \right\rfloor + 1 \\ &= \left\lceil \frac{z}{Q} \right\rceil \text{ (using Lemma 2)}\end{aligned}$$

It also implies that $z \bmod Q \neq 0$, which means:

$$\begin{aligned}& 0 < z \bmod Q < Q \\ \implies\ & 0 < Q - z \bmod Q < Q \\ \implies\ & 0 < f < Q\end{aligned}$$

□

Now suppose we know the maximum value n will take. We can then prove:

Theorem 10. *Suppose n is a natural number that can take value at most M, and Q is a fixed constant that is not a power of 2. Then, if $k = \lceil log_2(M) + log_2(Q) \rceil$ and $C = \lceil \frac{2^k}{Q} \rceil$ we have:*

$$\left\lfloor \frac{n}{Q} \right\rfloor = \left\lfloor \frac{n}{2^k} \times C \right\rfloor$$

Proof. By definition, $\frac{n}{Q} = \lfloor \frac{n}{Q} \rfloor + \epsilon$ where $\epsilon \in [0, 1)$, and since $Q \nmid 2^k$, Lemma 14 lets us express C as $\frac{2^k+f}{Q}$. Therefore:

$$\begin{aligned}\left\lfloor \frac{n}{2^k} \times C \right\rfloor &= \left\lfloor \frac{n}{2^k} \left(\frac{2^k}{Q} + \frac{f}{Q} \right) \right\rfloor \\ &= \left\lfloor \frac{n}{Q} + \frac{n}{2^k} \cdot \frac{f}{Q} \right\rfloor \\ &= \left\lfloor \left\lfloor \frac{n}{Q} \right\rfloor + \epsilon + \frac{n}{2^k} \cdot \frac{f}{Q} \right\rfloor\end{aligned}$$

Lemma 14 tells us that $\frac{f}{Q} < 1$. Using this along with the fact that $n < M$ and that $log_2(M) + log_2(Q) \leq \lceil log_2(M) + log_2(Q) \rceil$ we have:

$$
\begin{aligned}
2^{\lceil log_2(M)+log_2(Q)\rceil} &\geq 2^{log_2(M)+log_2(Q)} \\
\implies 2^k &\geq MQ \\
\implies \frac{1}{2^k} &\leq \frac{1}{MQ} \\
\implies \frac{n}{2^k} &< \frac{1}{Q} \\
\implies \frac{n}{2^k} \cdot \frac{f}{Q} &< \frac{1}{Q}
\end{aligned}
$$

Since, by Lemma 13, ϵ can be at most $\frac{Q-1}{Q}$:

$$
\begin{aligned}
\epsilon + \frac{n}{2^k} \cdot \frac{f}{Q} &< \frac{Q-1}{Q} + \frac{1}{Q} \\
\implies \epsilon + \frac{n}{2^k} \cdot \frac{f}{Q} &< 1
\end{aligned}
$$

And so, $\left\lfloor \left\lfloor \frac{n}{Q} \right\rfloor + \epsilon + \frac{n}{2^k} \cdot \frac{f}{Q} \right\rfloor = \left\lfloor \left\lfloor \frac{n}{Q} \right\rfloor \right\rfloor = \left\lfloor \frac{n}{Q} \right\rfloor$.

□

6.1 Ciphertext coefficient compression in Kyber

To perform ciphertext compression, Kyber implementations have to compute the value $\left\lfloor \frac{2^d}{Q} \cdot x + \frac{1}{2} \right\rfloor$. This is equivalent to:

$$
\begin{aligned}
&= \left\lfloor \frac{2^d}{Q} \cdot x + \frac{\frac{Q}{2}}{Q} \right\rfloor \\
&= \left\lfloor \frac{2^d \cdot x + \frac{Q}{2}}{Q} \right\rfloor \\
&= \left\lfloor \frac{2^d \cdot x + \left\lfloor \frac{Q}{2} \right\rfloor}{Q} \right\rfloor \text{ (using Lemma 3)}
\end{aligned}
$$

Moreover, $Q = 3329$ (and $\lfloor \frac{Q}{2} \rfloor = 1664$), d can take a maximum value of 11, and $0 \leq x < Q$.

If we let $n = 2^d \cdot x + \left\lfloor \frac{Q}{2} \right\rfloor$, n can take a maximum value of:

$$2^{11} \cdot 3328 + 1664 = 6,817,408$$

And so, using Theorem 10, we obtain $\lfloor \frac{n}{Q} \rfloor$ by computing $\lfloor \frac{nC}{2^k} \rfloor$ where $k = \lceil log_2(6,817,408) + log_2(3329) \rceil = 35$ and $C = \left\lceil \frac{2^k}{Q} \right\rceil = \left\lceil \frac{2^{35}}{3329} \right\rceil = 10,321,340$.

It is this expression that the `libcrux` implementation of Kyber computes:

```
pub(super) fn compress_ciphertext_coefficient(coefficient_bits: u8, fe: u16)
↪  -> FieldElement {
    ...
    ...
    let mut compressed = (fe as u64) << coefficient_bits;
    compressed += 1664 as u64;

    compressed *= 10_321_340;
    compressed >>= 35;
    ...

}
```

References

[1] Hanno Becker et al. "Neon NTT: Faster Dilithium, Kyber, and Saber on Cortex-A72 and Apple M1". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2022.1 (Nov. 2021), pp. 221–244. DOI: 10.46586/tches.v2022.i1.221-244. URL: https://tches.iacr.org/index.php/TCHES/article/view/9295.

[2] *Labor of Division (Episode I)*. https://ridiculousfish.com/blog/posts/labor-of-division-episode-i.html. Accessed: 23 January 2023.