# Mathematical Background for Residue Number Coding Obfuscation

R.Tiella

July 22, 2015

**Abstract**

Residue Number Coding Obfuscation is formally defined in terms of modular arithmetic. The present report is an attempt of comprehensively collect the theoretical facts this obfuscation technique is based on.

## 1 Background in Abstract Algebra

Informally an algebraic structure, is a set $S$ with a finite number of operations defined.

**Definition 1** (Monoid)**.** *An algebraic structure $(M, \cdot, 1)$ is a monoid if:*

1. *(associative rule) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$, for all $a, b, c \in M$*
2. *(null element) $a \cdot 1 = 1 \cdot a = a$, for all $a \in M$*

**Definition 2** (Group)**.** *An algebraic structure $(G, \cdot, 1)$ is a group if:*

1. *$G$ is monoid*
2. *(every element has an inverse): for all $a \in G$, exists $a' \in G$ such that $a \cdot a' = a' \cdot a = 1$*

**Definition 3** (Commutative Group)**.** *A group $(G, \cdot, 1)$ is commutative if:*

1. *(commutativity) for all $a, b \in G$, $a \cdot b = b \cdot a$*

**Definition 4** (Ring)**.** *An algebraic structure $(R, +, \cdot, 0, 1)$ is a ring if:*

1. *$(R, +, 0)$ is a commutative group*
2. *$(R, \cdot, 1)$ is a monoid*
3. *distributive laws hold:*
   (a) $(a + b) \cdot c = a \cdot c + b \cdot c$
   (b) $a \cdot (b + c) = a \cdot b + a \cdot c$

**Definition 5** (Commutative Ring)**.** *A ring $(R, +, \cdot, 0, 1)$ is commutative if:*

1. $(R, \cdot, 1)$ *is a commutative monoid.*

**Definition 6** (Ring Homomorphism)**.** *Given two rings $R$ and $R'$, a map $\varphi : R \to R'$ is a ring homomorphism if:*

1. $\varphi(r_1 + r_2) = \varphi(r_1) +' \varphi(r_2)$
2. $\varphi(r_1 \cdot r_2) = \varphi(r_1) \cdot' \varphi(r_2)$
3. $\varphi(0) = 0'$
4. $\varphi(1) = 1'$

# 2 Basic facts about $\mathbb{Z}$

**Definition** $\mathbb{Z}$ is the set of integer numbers, namely $\{0, 1, 2, ..., -1, -2, ...\}$.

**Proposition** $\mathbb{Z}$ with the usual operations of addition and multiplication is a commutative ring.

**Definition** Given a number $m \in \mathbb{Z}$, we can define the following relation on $\mathbb{Z}$:

$$x \equiv_m y \text{ iff } m|(x - y)$$

**Proposition** $\equiv_m$ is an equivalence relation:

1. $m|0$ so $m|(x - x)$
2. if $m|(x - y)$ then $m|(-(y - x))$, that is to say $m|(y - x)$
3. if $m|(x - y)$ and $m|(y - z)$ then $y = km + x$ so $m|(km + x - z)$ thus $m|(x - z)$

We also write $x \equiv y\ (m)$. We also drop $m$ as in $x \equiv y$, if doing so doesn't create confusion in the reader.

**Definition 7** (Congruence relation)**.** *A relation $x \sim y$ on the elements of a ring $R$ is said "a congruence relation" if it satisfies:*

- *if $x_1 \sim y_1$ and $x_2 \sim y_2$ then $x_1 + x_2 \sim y_1 + y_2$*
- *if $x_1 \sim y_1$ and $x_2 \sim y_2$ then $x_1 \cdot x_2 \sim y_1 \cdot y_2$*

**Proposition** $\equiv$ is a congruence relation.

**Definition** Given a $x \in \mathbb{Z}$, the congruence class of $x$ is:

$$[x]_m = \{y | x \equiv_m y\} = \{x + km | k \in \mathbb{Z}\}$$

**Proposition** The set of congruence modulo $m$ classes, i.e., $\mathbb{Z}_m = \{[x]_m | x \in \mathbb{Z}\}$. with the following operations:

1. $[x]_m +' [y]_m = [x + y]_m$
2. $[x]_m \cdot' [y]_m = [x \cdot y]_m$

and identities $[0]_m$ and $[1]_m$ is a commutative ring.

**Generalization** The space $\mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_n}$ with per-component addition and multiplication:

$$([x]_{m_1}, [x]_{m_2}, ..., [x]_{m_n}) +' ([y]_{m_1}, [y]_{m_2}, ..., [y]_{m_n}) = ([x+y]_{m_1}, [x+y]_{m_2}, ..., [x+y]_{m_n})$$

$$([x]_{m_1}, [x]_{m_2}, ..., [x]_{m_n}) \cdot' ([y]_{m_1}, [y]_{m_2}, ..., [y]_{m_n}) = ([x \cdot y]_{m_1}, [x \cdot y]_{m_2}, ..., [x \cdot y]_{m_n})$$

with the identity element for the addition:

$$([0]_{m_1}, [0]_{m_2}, ..., [0]_{m_n})$$

and the identity element for the multiplication:

$$([1]_{m_1}, [1]_{m_2}, ..., [1]_{m_n})$$

is a commutative "ring".

**The RNC homomorphism** Given $m_1, m_2, ..., m_n \in \mathbb{Z}$, we define the transformation:

$$\varphi : \mathbb{Z} \to \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_n}$$

as

$$\varphi(x) = ([x]_{m_1}, [x]_{m_2}, ..., [x]_{m_n})$$

The map $\varphi$ is a ring homomorphism, and, please note, it is clearly not injective.

**Definition 8** (Greatest Common Divisor, (g.c.d.))**.** *Given two integers $a, b$, the greatest common divisor* $\gcd(a, b)$ *of $a$ and $b$ is an integer number which satisfies:*

- $\gcd(a, b)|a$ *and* $\gcd(a, b)|b$
- *if $m|a$ and $m|b$ then* $m|\gcd(a, b)$

**Theorem 1** (Extended Euclid's Algorithm)**.** *Given two integers $a$ and $b$ exist two integers $x$ and $y$ so that*

$$\gcd(a, b) = ax + by$$

*Proof.* Consider three sequences $(r_i)$, $(x_i)$, and $(y_i)$ defined by the following iterative formulas:

$$\begin{array}{ll} r_1 = a & r_2 = b \\ x_1 = 1 & x_2 = 0 \\ y_1 = 0 & y_2 = 1 \end{array}$$

$$\begin{array}{c} r_{i-2} = r_{i-1}q_i + r_i, \text{ with } q_i \in \mathbb{Z} \text{ and } |r_i| < |r_{i-1}| \\ x_{i-2} = x_{i-1}q_i + x_i \\ y_{i-2} = y_{i-1}q_i + y_i \end{array}$$

It can be proven that the sequence $(r_n)$ reaches zero and the last $r_n \neq 0$ is $\gcd(a, b)$ while $x_n, y_n$ satisfy the equation above. □

The previous theorem justifies the following:

**Algorithm 1** Extended Euclid's Algorithm

```
(d,x,y) = function EGCD(a,b):
if b = 0 then
  return (a, 1, 0)
else
  (d,x1,y1) = EGCD(b, mod(a, b))
  return (d, y1, x1 − y1 ∗ (a/b))
end if
```

**Lemma 1.** *If $a$ and $m$ are coprime, the equation:*

$$ax \equiv 1(m)$$

*has a unique solution modulo $m$.*

*Proof.* Directly from the Euclidean Algorithm, exists $u$ and $v$ such that:

$$ua + vm = 1$$

i.e.,

$$au = 1 - vm$$

i.e.,

$$au \equiv 1(m)$$

so $u$ is a solution of the equation. $u' = u + km$ is also a solution. □

**Theorem 2** (Chinese Remainder Theorem)**.** *Let $m_1, m_2, ..., m_n$ be pair-wise coprime integer numbers, the system of congruence equations:*

$$\begin{cases} x \equiv a_1 \quad (m_1) \\ x \equiv a_2 \quad (m_2) \\ \quad \vdots \\ x \equiv a_n \quad (m_n) \end{cases}$$

*has a unique solution module $M$, where $M = \prod_{i=1,...,n} m_i$. In particular the solution is given by:*

$$x \equiv a_1 M_1 b_1 + a_2 M_2 b_2 + ... + a_n M_n b_n \ (M)$$

*with* $M_i = M/m_i$ *and* $M_k b_k \equiv 1(m_k)$.

*Proof.*

(a) [The solution exists]: as $gdc(M_k, m_k) = 1$, equation $M_k b_k \equiv 1(m_k)$ has a unique solution module $m_k$, then the term $a_k M_k b_k \equiv a_k(m_k)$ while $a_k M_k b_k \equiv 0(m_j)$ if $k \neq j$ (because $m_j | M_k$ if $k \neq j$).

(b) [The solution is unique]: if $y$ is another solution of the system, then $x - y \equiv 0(m_k)$ for all $k$, then $x - y \equiv 0(M)$.

□

If we restrict $\varphi$ to an interval $I$ of size $M$ e.g., $I = [0, M-1]$, the previous theorem guarantees $\varphi_{|_I}$ is invertible.

# 3 Residue Number Coding

The C.R.T. gives us a tool to make the RNC homomorphism injective, that is to say, an isomorphism.

In what follows we restrict the number of components to 2, i.e., we assume $n = 2$.

**Encoding** A base $(m_1, m_2)$ is chosen satisfying $\gcd(m_1, m_2) = 1$. The RNC encoding, defined by the RNC homomorphism, simplifies to:

$$e(\nu) = (\mathrm{mod}(\nu, m_1) + r_1 m_1, \mathrm{mod}(\nu, m_2) + r_2 m_2) = (\mu_1, \mu_2)$$

where $r_i \in \mathbb{Z}$ are randomly chosen.

**Decoding** The decoding function is defined as:

$$d(\mu_1, \mu_2) = \nu$$

where $\nu$ satisfies:

$$\begin{cases} \nu & \equiv & \mu_1 & (m_1) \\ \nu & \equiv & \mu_2 & (m_2) \end{cases}$$

Applying the C.R.T.,

$$\nu \equiv \mu_1 M_1 b_1 + \mu_2 M_2 b_2 \ (M)$$

with $M_1 = M/m_1 = m_2$ and $M_2 = M/m_2 = m_1$ and each $b_k$, $k = 1, 2$ satisfying the equation $M_k b_k \equiv 1(m_k)$, so

$$\nu = \mu_1 m_2 \mathtt{EGCD}_2(m_2, m_1) + \mu_2 m_1 \mathtt{EGCD}_2(m_1, m_2) + kM$$

with $k$ such that $\mu \in I$. Please note that terms like $\mathtt{EGCD}_2(m_i, m_j)$ depend on values chosen for the base $\{m_\}$ only , particularly don't depend on $\nu$ or $(\mu_1, \mu_2)$.

Given value $\nu \in \mathbb{Z}$ the representant of $[\nu]_m$ in $[0, ..., m-1]$ is easily found as: mod ( mod $(\nu, m) + m, m)$).

**Example of application:** RNC allows to obfuscate variables and to evaluate expressions involving addition, subtraction, and multiplication without the need of decoding back as in the default encoding transformation. The original code is on the left side of the following frame while the obfuscated one is on the right side.

```
#include <stdio.h>

int main(int argc, char * argv[]) {

        int x = 12;
        int y = 7;

        int z = x+y;
        int w = x*y;

        printf("z=%d, w=%d\n",z,w);

}
```

```
#include <stdio.h>

#define m1 14
#define m2 15

#define M (m1*m2)
#define M1 (M/m1)
#define M2 (M/m2)

int decode(int w1, int w2);

int main(int argc, char * argv[]) {

int x1 = 1965;
int x2 = 1973;

int y1 = 1433;
int y2 = 2634;

int z1 = x1+y1;
int z2 = x2+y2;

int w1 = x1*y1;
int w2 = x2*y2;

printf("z=%d, w=%d\n",
              decode(z1,z2),decode(w1,w2));

}
```

In the following frame the helping function used in the transformed code.

```
 void e_gcd(int a, int b, int *x, int *y)
{
            int temp;

            if (b > a) {
                e_gcd(b, a, y, x);
    }

            if (b == 0) {
                *x = 1;
                *y = 0;

            } else {

            e_gcd(b, a%b, x, y);

            temp = *x;
            *x = *y;
            *y = temp - (a / b) * *y;

    }
}
```

```
int decode(int w1, int w2) {
            int x1, x2, y1, y2;
            int result;


            e_gcd(M1, m1, &x1, &y1);
            e_gcd(M2, m2, &x2, &y2);

            result = (((w1 * x1 * M1  +
                        w2 * x2 * M2) % M) + M) % M;

            return result;
}
```