EasyCrypt Library in Jasmin

v 1.0

Ji, Yong-hyeon

(hacker3740@kookmin.ac.kr)

Department of Information Security, Cryptology, and Mathematics

College of Science and Technology

Kookmin University

January 3, 2025

File Structure

- jasmin
  - compiler/
  - eclib/
    - `AES.ec`
    - `JArray.ec`
    - `JLeakage.ec`
    - `JMemory.ec`
    - `JModel.ec`
    - `JModel_common.ec`
    - `JModel_m4.ec`
    - `JModel_x86.ec`
    - `JUtils.ec`
    - `JWord.ec`
    - `JWord_array.ec`
    - `Jslh.ec`
    - `Makefile`
    - `easycrypt.project`
    - `tests.config`

Copyright

Changelog

v1.0	2025-01-03	Initial release:

Contents

1 JUtils

```
require import AllCore IntDiv List Bool StdOrder.
        import IntOrder.
```

https://github.com/EasyCrypt/easycrypt
easycrypt/theories/core/AllCore.ec
easycrypt/theories/core/Bool.ec
easycrypt/theories/algebra/IntDiv.ec
easycrypt/theories/algebra/StdOrder.ec
easycrypt/theories/datatypes/List.ec

LEMMA: modz_comp

```
lemma modz_cmp m d : 0 < d => 0 <= m %% d < d.
proof. smt (edivzP). qed.
```

Statement. For two integers m and $d > 0$, the remainder of m divided by d satisfies:

$$0 \leq m \bmod d < d.$$

Analysis. This property follows directly from the division algorithm:

$$m = q \cdot d + r, \quad 0 \leq r < d$$

where $q = \lfloor m/d \rfloor$ and $r = m \bmod q$.

Proof Tactics. SMT solver with the pre-proved property `edivzP` (in `IntDiv`).

LEMMA: divz_cmp

```
lemma divz_cmp d i n : 0 < d => 0 <= i < n * d => 0 <= i %/ d < n.
proof.
  by move=> hd [hi1 hi2]; rewrite divz_ge0 // hi1 /= ltz_divLR.
qed.
```

Statement. For integers d, i, n where $d > 0$ and $0 \leq i < n \cdot d$, the integer division satisfies

$$0 \leq \frac{i}{d} < n.$$

Analysis. TBA

Proof Tactics. TBA

LEMMA: mulz_cmp_r

```
lemma mulz_cmp_r i m r : 0 < m => 0 <= i < r => 0 <= i * m < r * m.
proof.
  move=> h0m [h0i hir]; rewrite IntOrder.divr_ge0 //=; 1: by apply ltzW.
  by rewrite IntOrder.ltr_pmul2r.
qed.
```

Statement. TBA

Analysis. TBA

Proof Tactics. TBA

LEMMA: cmpW

```
lemma cmpW i d : 0 <= i < d => 0 <= i <= d.
proof. by move=> [h1 h2];split => // ?;apply ltzW. qed.
```

Statement. TBA

Analysis. TBA

Proof Tactics. TBA

LEMMA: `le_modz`

```
lemma le_modz m d : 0 <= m => m %% d <= m.
proof.
  move=> hm.
  have [ ->| [] hd]: d = 0 \/ d < 0 \/ 0 < d by smt().
  + by rewrite modz0.
  + by rewrite -modzN {2}(divz_eq m (-d)); smt (divz_ge0).
  by rewrite {2}(divz_eq m d); smt (divz_ge0).
qed.
```

Statement. TBA

Analysis. TBA

Proof Tactics. TBA

2 JArray

References

A Algebra

A.1 IntDiv

```
op euclidef (m d : int) (qr : int * int) =
     m = qr.`1 * d + qr.`2
  /\ (d <> 0 => 0 <= qr.`2 < `|d|).

op edivn (m d : int) =
  if (d < 0 \/ m < 0) then (0, 0) else
    if d = 0 then (0, m) else choiceb (euclidef m d) (0, 0)
  axiomatized by edivn_def.

op edivz (m d : int) =
  let (q, r) =
    if 0 <= m then edivn m `|d| else
      let (q, r) = edivn (-(m+1)) `|d| in
      (- (q + 1), `|d| - 1 - r)
    in (signz d * q, r)
  axiomatized by edivz_def.

abbrev (%/) (m d : int) = (edivz m d).`1.
abbrev (%%) (m d : int) = (edivz m d).`2.
```

```
lemma edivzP (m d : int) :
  m = (m %/ d) * d + (m %% d) /\ (d <> 0 => 0 <= m %% d < `|d|).
proof. by case: (edivzP_r m d). qed.
```