CLIP: A Cryptographic Language with Irritating Parentheses

Yi-Hsiu Chen(yc2796)
Wei Duan(wd2214)
Introduction

- Motivation
  - Cryptographers

- Language features
  - Functional (Lisp)
  - Big number (GMP)
  - Bit sequence

- Cryptosystem
  - Public Key (RSA, El Gamel...)
  - Private Key (AES...)
Language Tutorial

(defvar | defun | expression)*

defvar i:int = 5;
defvar bs:bit#7 = '1011010;
defvar s:string = "Hello World";
defvar n:int[2][3] = {{1 2 4} {3 5 7}};
defun square:int n:int = (* n n);
(+ (square 3) 2)
### Built-in function

<table>
<thead>
<tr>
<th>For big number</th>
<th>For logic operation</th>
<th>For comparison operation</th>
<th>For conversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>+, -, *, /, mod, pow, inverse, is-prime, next-prime</td>
<td>and, or, not, (^{(xor)})</td>
<td>less, greater, leq, geq, eq, neq, and, or, not</td>
<td>int-of-bits, bits-of-int, string-of-bits, bits-of-string</td>
</tr>
<tr>
<td>For bits number</td>
<td>For vector operation</td>
<td>Miscellaneous</td>
<td></td>
</tr>
<tr>
<td>zero, rand, pad, flip, flip-bit</td>
<td>group, merge, make-vector, transpose</td>
<td>let, map, reduce, lambda</td>
<td></td>
</tr>
</tbody>
</table>
Example Program

defun gcd:int a:int b:int=
(if (eq a b)
  a
  (if (greater a b)
    (gcd (− a b) b)
    (gcd a (− b a))))

(gcd 5 15)  ~~ Print 5
Example Program

(defun theta:bit#64[5][5] block:bit#64[5][5] =
  (let <col-par:bit#320 (reduce ^ (map merge block))>
    (map (lambda <row:bit#320>
          (group (^ row (>>> col-par 64) (<<< col-par 63)) 64))
       (map merge block)));

(let <variable:type expression1> expression2)

(map function vector)

(reduce function vector)

(lambda <argument:type> expression)
Architecture
Demo

- 1. GCD
- 2. SHA-3
- 3. RSA
SHA-3

\((\theta \rightarrow \rho \rightarrow \pi \rightarrow \chi \rightarrow \iota) \times 24\)

<table>
<thead>
<tr>
<th>64 bits</th>
<th>1010...10</th>
<th>0000...11</th>
<th>0101...10</th>
<th>1110...11</th>
<th>0110...00</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101...01</td>
<td>1001...01</td>
<td>0101...00</td>
<td>1111...11</td>
<td>0111...11</td>
<td></td>
</tr>
<tr>
<td>0000...00</td>
<td>1111...11</td>
<td>0100...01</td>
<td>0100...11</td>
<td>0011...11</td>
<td></td>
</tr>
<tr>
<td>1000...11</td>
<td>0111...01</td>
<td>1001...11</td>
<td>0000...00</td>
<td>1101...10</td>
<td></td>
</tr>
<tr>
<td>1111...00</td>
<td>0010...10</td>
<td>0100...11</td>
<td>0000...00</td>
<td>1010...10</td>
<td></td>
</tr>
</tbody>
</table>

| ⊕ | | | | | |
|---|---|---|---|---|
| 1000...00 | 0001...10 | 1001...11 | 0101...11 | 0011...00 |
| 0011...00 | 1000...00 | 0001...10 | 1001...11 | 0101...11 |
| 0000...00 | 0100...11 | 1010...11 | 1001...10 | 0100...00 |
RSA

\[ n = p \cdot q \]
\[ N = \phi(n) = (p - 1)(q - 1) \]
\[ e = 65537 \]

Public: n, e

Private: N, d

\[ d = e^{-1} \mod N \]

\[ m = c^e \mod n \]

\[ c = m^d \mod n \]