Magpie:一种高安全的轻量级分组密码算法

李 浪[1,2],李肯立[2],贺位位[1],邹 祎[1],刘波涛[1]
(1. 衡阳师范学院计算机科学系,湖南衡阳 421008;2. 湖南大学信息科学与工程学院,湖南长沙 410082)

摘 要: 论文提出了一种新的高安全轻量级密码算法,命名为 Magpie. Magpie 是基于 SPN 结构,分组长度为 64 位,密钥长度为 96 位,包含 32 轮运算. Magpie 密码算法包括两个部分:运算部分和控制部分. 运算部分,每轮运算包括五个基本运算模块:常数加,S 盒变换,行移位,列混合,轮密钥加. 控制部分,将密钥的第 65 位到 96 位作为 Magpie 加密算法的控制信号,其中密钥第 65 位到第 80 位作为 S 盒变换控制信号,第 81 位到第 96 位值作为列混合,行移位变换和每轮运算的控制信号. 在 Xilinx Virtex-5 FPGA 上实现面积仅为 10679 Slices,加密速率为 6.4869Gb/s.

关键词: 轻量级密码; 分组密码; FPGA 实现
中图分类号: TP309 **文献标识码:** A **文章编号:** 0372-2112 (2017)10-2521-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2017.10.029

Magpie:a High-Security Lightweight Block Cipher

LI Lang[1,2],LI Ken-li[2],HE Wei-wei[1],ZOU Yi[1],LIU Bo-tao[1]
(1. *Department of Computer Science,Hengyang Normal University,Hengyang Hunan* 421008,*China*;
2 *College of Information Science and Engineering,Hunan University,Changsha,Hunan* 410082,*China*)

Abstract: We present,so called,Magpie which is a new high-security lightweight block cipher. The block size of Magpie is 64 bits and the key size is 96 bits. It employs a SPN structure and consists of 32 rounds. Magpie encryption algorithm includes two parts:operation part and control part. Each operational round includes five basic modules:AddConstants,SubCells,ShiftRows,MixColumns,AddRoundKey. The control part is the key of 65 to 96 bits. 65 to 80 bits of key control the SubCells. 81 to 96 bits of key control the MixColumns and the ShiftRows. The control signal can control the order of the module operation. The Xilinx Virtex-5 FPGA hardware area of Magpie requires about 10679 slices and the throughout rate is 6.4869Gb/s.

Key words: lightweight cryptography;block cipher;FPGA implementation

1 引言

随着物联网的深入应用,如何在资源受限的智能卡上实现安全高效加密成为重要研究课题. 分组密码算法由于其独有的优势,使其成为轻量级密码算法的首要结构,在过去几年,提出了一系列轻量级分组加密算法如:PRESENT[1],PUFFIN[2],MIBS[3],PRINTCipher[4],LED[5],EPCBC[6],LBlock[7],Piccolo[8],Klein[9],Twine[10],PRINCE[11]等. 轻量级分组密码算法的设计一般应该满足以下要求:低资源占用,低功耗,相同的加解密结构,较高的安全性,较好的加密吞吐率. 如何设计一个安全高效的轻量级分组密码算法目前仍然是一个具有挑战性的问题,如何在资源受限的智能卡上加密并满足安全、高性能、小面积、低功耗等要求,这仍然比较难以做到.

本文设计并实现了一个较高安全性的轻量级分组密码算法,取名为 Magpie.

2 Magpie 密码算法设计原理

Magpie 密码算法由二部分组成:运算部分和控制部分.

收稿日期:2013-12-12;修回日期:2016-03-20;责任编辑:郭游
基金项目:国家自然科学基金(No.61572174);湖南省自然科学基金(No.2015JJ4011);湖南省教育厅科研重点基金(No.15A029);湖南省科技计划项目(No.2016TP1020)

2.1 Magpie 运算流程

Magpie 密码算法属于分组密码算法,分组长度为 64 位,密钥长度固定为 96 位,加密运算 32 轮后输出 64 位密文. 32 轮运算可以保证 Magpie 密码算法抗差分和线性攻击. 每轮运算部分包括五个基本运算模块:常数加(AddConstants),S 盒变换(SubCells),行移位(ShiftRows),列混合(MixColumns),轮密钥加(AddRoundKey).

Magpie 密码算法的具体运算流程如图 1 所示. 从图 1 中可以得到,密钥更新(Updatekey)操作是对输入的密钥复用轮操作中 SubCells,从而达到密钥混淆与扩散的目的,并把输出作为后续 Updatekey 的输入. 具体操作为 96 位密钥分为两部分:第 1 位到 64 位密钥做轮加密运算,第 65 位到 96 位密钥做轮控制,Updatekey 操作是对第 1 位到 64 位密钥进行轮运算操作中 S 盒变换. 输出的新密钥进行轮加密运算与下一轮密钥更新.

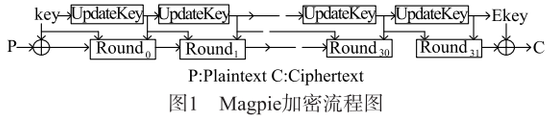


图1 Magpie加密流程图

2.2 Magpie 密码算法内部结构

Magpie 密码算法是基于 SPN 结构,轮运算结构如图 2 所示,共有 32 轮相同的轮运算. Magpie 密码算法 $Round_i$($0 <= i < 32$)内部结构主要有:引用中间状态(state),key[64]到 key[79]为 S 盒控制信号,控制信号 select(key[80]到 key[95])为行移位变换与列混合的控制信号,同时控制整个算法加密顺序与轮数. 每轮运算中五个基本运算的顺序:

当 select = 1,模块运行的顺序为,

AddConstants→SubCells→ShiftRows→MixColumns→AddRoundKey;

当 select = 0,模块运行的顺序为,

AddRoundKey→MixColumns→ShiftRows→SubCells→AddConstants.

以附录 A 中的测试向量举例说明:明文(用 16 进制表示):0123_4567_89AB_CDE

F;密钥(用 16 进制表示):0123_4567_89A

B_CDEF_0123_4567.

密钥 key[64]到 key[79]为 S 盒替换的控制信号:$(0123)_{16} = (0000\_0001\_0010\_0011)_2$;密钥最后 16 位为 select 控制信号:

$(4567)_{16} = (0100\_0101\_0110\_0111)_2$.

这些 0、1 都是从输入密钥中取出来做控制的二进制数,密钥 key[64]到 key[79]为 S 盒替换的控制信号,控制着明文的 S 盒变换与密钥的 S 盒变换. select 控制信号,决定 Magpie 密码算法的行移位与列混合的控制信号,同时控制整个算法加密顺序与轮数. 由于控制信号是从密钥的相应位决定,而密钥又是用户随机自定的,这样整个算法加密运算模块顺序是不固定的,从而增加攻击者的破解难度.

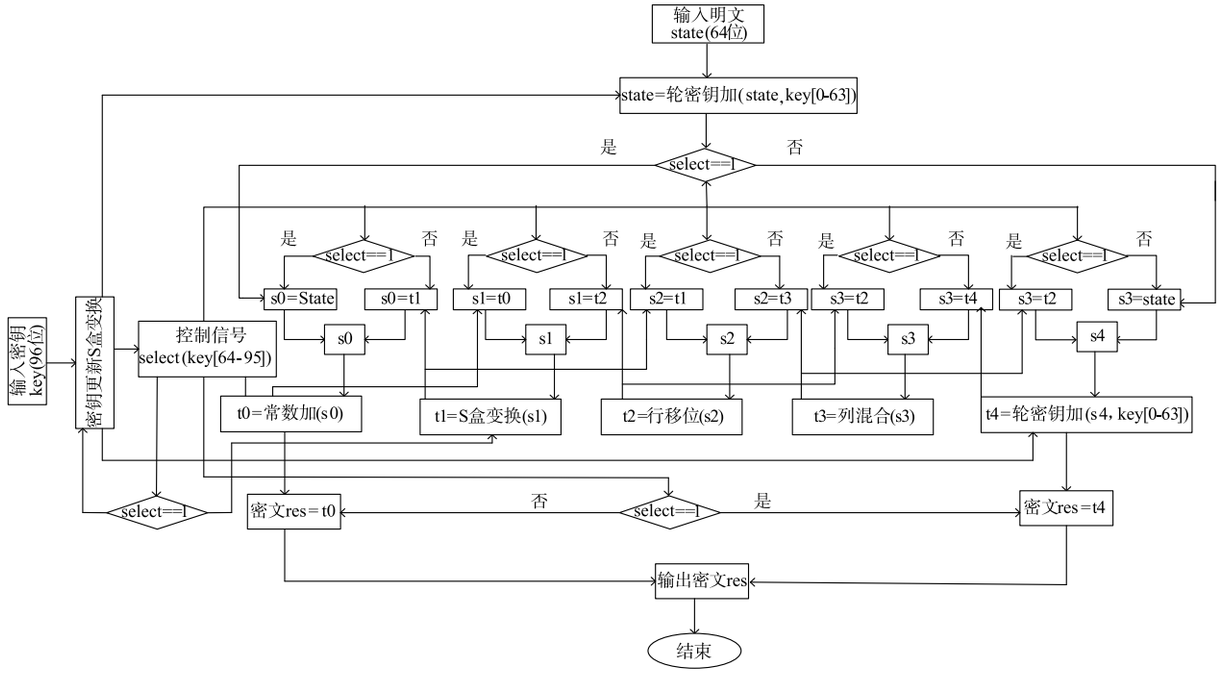


图2 Magpie密码算法轮运算结构图

3　Magpie 加密算法实现

3.1　SubCells 中 S 盒构造方法

Magpie 的 S 盒构造是在 PRESENT 密码算法 S 盒的基础上进行修改,构造一个二维数组,使其满足如下关系:

$$x = sbox[1][sbox[0][x]];$$
$$x = sbox[0][sbox[1][x]];$$

通过上面的构造公式,给出其中 Sbox[2][16]的值如下:

```
Sbox[2][16] = {
5,14,15,8,12,1,2,13,11,4,6,3,0,7,9,10,
12,5,6,11,9,0,10,13,3,14,15,8,4,7,1,2
};
```

3.2　MixColumns 中列矩阵构造方法

Magpie 密码算法包括两个矩阵. 第一个矩阵构造如下[5].

```
const byte MixColMatrix[16] = {
4,  1,  2,  2,
8,  6,  5,  6,
11,14,  10,  9,
2,  2,  15,  11
};
```

由上可以求出 MixColMatrix 的逆矩阵如下:

```
const byte r_MixColMatrix[16] = {
12,  12,  13,  4,
3,   8,   4,   5 ,
7,   6,   2,   14 ,
13,  9,   9,   13
};
```

将上面的 MixColMatrix[16]及其逆矩阵 r_MixColMatrix[16]组合到一个矩阵里,这样加解密就可以复用,排列如下:

```
byte Matrix1[32] = {
4,  12,  1,  12,  2,  13,  2,  4,
8,  3,   6,  8,   5,  4,   6,  5,
11,  7,  14,  6,  10,  2,  9,  14,
2,  13,  2,  9,   15,  9,  11,13
};
```

3.3　Magpie 轮运算

明文(Plaintext)为 64 位,将其中每 4 位记作一个 $m_i(0 <= i < 16)$;Plaintext = $m_0 \| m_1 \| \cdots \| m_{14} \| m_{15}$;同理将 96 位密钥 key 记作:key = $k_0 \| k_1 \| \cdots \| k_{22} \| k_{23}$. 加密中间状态命名为 state.

AddConstants:将 state 最左边八位和最右边八位与 RC[i](0 <= i < 32)异或. RC[i]的具体值如下:

```
byte    RC[RN] = {
    0x02,0x03,0x06,0x0A,
    0x3C,0x92,0xA3,0x61,
    0xA8,0xCD,0xFE,0x3B,
    0x2C,0x6E,0x25,0x6D,
    0x6D,0x25,0x6E,0x2C,
    0x3B,0xFE,0xCD,0xA8,
    0x61,0xA3,0x92,0x3C,
    0x0A,0x06,0x03,0x02
};
```

该数组有一个显著的特点是:RC[i] = RC[31 - i].

SubCells:将 state 记作:state = $S_0 \| S_1 \| \cdots \| S_{14} \| S_{15}$. 每轮运算定义为:

C_i = Sbox[key[64 + i]][S_i];(0 <= i < 16).

ShiftRows:该模块中有控制信号 select,将 state 看作一个 4 × 4 的矩阵,每个元素由 4 位二进制数组成,如果控制信号 select 等于 1,采用第一种方案:state 的第一行循环左移一个半字节,第二行循环左移 1 个字节,第三行循环左移半个字节,第四行保持不动. 如果控制信号 select 等于 0,采用第二种方案:state 的第一行循环右移一个半字节,第二行循环右移 1 个字节,第三行循环右移半个字节,第四行保持不动.

MixColumns:该模块中有控制信号 select,将明文 state 看作一个 4 × 4 的矩阵,每个元素有 4 位,当 state 和 MixColumns 相乘时,如果控制信号 select 等于 1,state × r_MixColMatrix;如果控制信号 select 等于 0,state × MixColMatrix.

在 MixColMatrix 与 r_MixColMatrix 中,还需要根据 select 信号的值选择矩阵中的相应元素参与运算,有如下关系:

当 select 信号为 1 时,选出矩阵 r_MixColMatrix 中的元素:12,12,13,4,3,8,4,5 ,7,6,2,14 ,13,9,9,将这 16 个元素组成一个 4 × 4 的矩阵和 state 进行有限域 $(X^4 + X + 1)$ 乘法运算;

当 select 信号为 0 时,选出矩阵 MixColMatrix 中的元素:4,1,2,2,8,6,5,6,11,14,10,9,2,2,15,11;将这 16 个元素组成一个 4 × 4 的矩阵和 state 进行有限域 $(X^4 + X + 1)$ 乘法运算.

AddRoundKey:将 state 和密钥的最左边 64 位进行异或运算.

Round:模块中包括以上五个运算模块,密钥 key 的最后 16 位作为 round 控制信号,每一位信号控制连续两轮运算.

Updatekey:将密钥最左边的 64 位每轮做一次 S 盒变换,密钥 S 盒变换和明文 S 盒变换相同.

3.4 Magpie 加密算法描述

用 C 语言对 Magpie 密码算法中加密运算描述如下.

算法 1 Magpie 加密运算

```
输入:Plaintext,密钥 key.
输出:密文 Ciphertxt
1:state = Plaintext;
2:AddRoundKey(key);
3:For(i=0;i<32;i++){
4:    if(key[81+(i/2)]==1){
5:        Round(state,key);
6:SubCells(key);
7:    }
8:else{
9:SubCells(key);
10:   Round(state,key);
11:}
12:}
13:Ciphertext = state;
```

4 Magpie 解密算法实现

Magpie 密码算法的优点之一就是解密完全复用加密模块. 具体操作如下:

第 1)步:将密钥最后 32 位二进制数取反(即 1 变成 0,0 变成 1);

第 2)步:将密钥最后 16 位二进制数倒置($k_{80}k_{81}k_{82}\cdots\cdots k_{93}k_{94}k_{95}\rightarrow k_{95}k_{94}k_{93}\cdots\cdots k_{82}k_{81}k_{80}$).

Magpie 加密算法运算部分中的逆运算:

1) AddConstants: RC 如下公式表示:RC[i] = RC[31 - i],可以看出 AddConstants 算法逆运算就是其本身.

2) SubCells: Sbox 数组有满足如下关系:

$x = sbox[1][sbox[0][x]]$;

$x = sbox[0][sbox[1][x]]$.

而 Sbox 第一个下标为密钥 key[64]到 key[79]控制,只需要将密钥 key[64]到 key[79]的控制信号取反,便能满足上述关系,从而实现 SubCells 的逆运算.

3) ShiftRows:同样是受控制信号 select 作用,控制信号 select 等于 1 与控制信号 select 等于 0 时进行交换,分别是控制行移位相反的运算.

4) MixColumns:其中矩阵 MixColumns 是通过 select 信号的值控制 MixColMatrix[16]与 r_MixColMatrix[16]两个不同的矩阵:

当 select 信号为 1 时,选出矩阵 r_MixColMatrix 中的元素:{12,12,13,4,3,8,4,5,7,6,2,14,13,9,9,13};

当 select 信号为 0 时,选出矩阵 MixColMatrix 中的元素:{4,1,2,2,8,6,5,6,11,14,10,9,2,2,15,11};

而这两个矩阵互为逆矩阵,因此只需要将密钥 key 的控制信号取反,便能实现该运算的逆运算.

5) AddRoundKey:在特征值为 2 的有限域中,异或的逆运算就是其本身.

将 Magpie 加密算法每轮加密记做 R_i(0 <= i < 32),则加密过程如下(其中下面 P 代表明文规定 Plaintext,C 代表密文 Ciphertext):

$P\rightarrow R_0(key[81])\rightarrow R_1(key[81])\rightarrow R_2(key[82])\rightarrow R_3(key[82])\cdots\rightarrow R_{28}(key[95])\rightarrow R_{29}(key[95])\rightarrow R_{30}(key[96])\rightarrow R_{31}(key[96])\rightarrow C$.

将密钥按照如下规则修改后则为解密运算:

1)将密钥最后 32 位二进制数取反(即 1 变成 0,0 变成 1);

2)将密钥最后 16 位二进制数倒置($k_{81}k_{82}k_{83\cdots}k_{94}k_{95}k_{96}\rightarrow k_{96}k_{95}k_{94\cdots}k_{83}k_{82}k_{81}$).

Magpie 解密运算过程如下(其中下面 P 代表明文规定 Plaintext,C 代表密文 Ciphertext):

$C\rightarrow R_0(key[96])\rightarrow R_1(key[96])\rightarrow R_2(key[95])\rightarrow R_3(key[95])\cdots\rightarrow R_{28}(key[82])\rightarrow R_{29}(key[82])\rightarrow R_{30}(key[81])\rightarrow R_{31}(key[81])\rightarrow P$.

5 安全性分析

在本节对 Magpie 密码算法的安全性进行分析.

5.1 差分攻击与线性攻击

对于 SPN 结构加密算法,S 为置换层,P 为扩散层. 加密算法抗差分与线性攻击主要根据 P 扩散层的分支数,决定着每轮活跃 S 盒的变化数.

引理 1 算法的 n×n 矩阵对应于的 P 扩散层分支数为 n+1,Magpie 算法的列混合为 4×4 矩阵变换,分支数为 5[12].

证明见参考文献[12].

定理 1 任何四轮 Magpie 算法的差分特征(differential characteristic)最少有 25 个活跃 S 盒.

证明:

输入 Magpie 算法扩散层数据中存在活动的半字节(半字节二进制不全为 0 数)称为一个活跃 S 盒. Magpie 算法的每四轮差分特征分析如下:其中 R_i 表示第 i 轮输出的差分特征(1 <= i <= 32).

Magpie 算法某一轮输入差分特征为一个活跃 S 盒,第一轮 R_1 为 1 个活跃 S 盒,第二轮 R_2 为 4 个活跃 S 盒,经过第三轮 Magpie 算法的轮变换,这 4 个活跃 S 盒扩散为 16 个活跃 S 盒,第三轮 R_3 为 16 个活跃 S 盒. 经过第四轮 Magpie 算法的变换,变为 4 个活跃 S 盒,第四轮 R_4 为 4 个活跃 S 盒.

通过 Magpie 算法四轮加密运算差分特征分析,四轮运算后,活跃 S 盒最少有 1 +4 +16 +4 =25.

不失一般性,如果某一轮输入两个甚至更多的差分特征,经过四轮变换直接产生最小数目活跃 S 盒将不低于 25. 所以,Magpie 算法的任何四轮变换差分特征有最小值 25 个活跃 S 盒.

证毕.

定理 2　任何四轮 Magpie 算法的线性特征(linear approximation)最少有 25 个活跃 S 盒.

证明:

由于 Magpie 算法的列混合为 4 × 4 矩阵变换保证了差分分支数与线性分支数为 5. 因此四轮变换的线性活跃 S 盒数最小为差分的活跃 S 盒数. 由定理 1 的证明中可知 Magpie 算法的任何四轮变换差分特征有最小值 25 个活跃 S 盒.

所以,任何四轮 Magpie 算法的线性特征(linear approximation)最少有 25 个活跃 S 盒.

证毕.

一个密码算法对抗差分与线性的强度分别表现在差分特征与线性特征的活跃 S 盒数量,上述算出的差分特征与线性特征活跃 S 盒是最少数量,根据确定轮的差分特征与线性特征值,评估 Magpie 算法的抗差分与线性的能力. Magpie 算法的 S 盒有最大的输出差分分配(differential distribution)与线性逼近(linear approximation)为 2^{-2};magpie 算法的 S 盒差分分配与线性近似分析为:最大输出差分达到为 $4 \div 16 = 2^{-2}$;最大线性近似达到$(2 \times 4 \div 16 - 1)^2 = 2^{-2}$.

结论 1 :Magpie 算法 32 轮差分特征值为 $(2^{-2})^{32 \times 25 \div 4} = 2^{-400}$.

结论 2 :Magpie 算法 32 轮线性特征值为 $(2^{-2})^{32 \times 25 \div 4} = 2^{-400}$.

结合 Magpie 算法中密钥控制加解密运算的顺序,算法的抗差分与线性攻击的能力将还要增强为 2^{16}倍.

通过上述的 Magpie 算法抗差分与线性攻击的分析,Magpie 算法对抗差分与线性攻击有绝对优势.

5.2　代数攻击

代数攻击也是密码算法设计时需要抵抗的一种基本攻击[13]. 针对 Magpie 中的 S 盒构造一个二维数组. 半字节的 S 盒可以由 GF(2)上的 21 个二次方程描述. 任何半字节四位 S 盒可以由至少 21 个这样的方程描述. 整个 Magpie 密码算法就可以通过 E = N × 21 个二次方程构造,二次方程由 V = N × 8 个变量描述,其中 N 是在 Magpie 加密算法和密钥调度中使用 S 盒的数量[14]. 在 Magpie 算法中:N = 32 × 16 + 16,因此整个系统由 11088 个二次方程、4224 个变量组成,这是一个典型的超定多变量高次方程组,即方程数多于变量数,求解一个这样的超定多变量二次方程组的问题是 NP Hard,复杂度是关于 n 的指数. 与 AES 加密算法相比,Magpie 的方程数和变量个数更多,目前在合理的时间和复杂度内代数攻击仍然不能获取 AES 全部密钥,因此,代数攻击对 Magpie 不构成威胁.

6　Magpie 密码算法性能分析

对 Magpie 密码算法进行了 FPGA 硬件实现验证其性能,通过 ISE 13. 2 对 Magpie 综合下载进行性能分析,FPGA 型号为 Xilinx virtex-5 LX50T. 图 3 是 Magpie 下载到 FPGA 上的 Slice 占用实验数据直接截图.

```
Number of errors:      0
Number of warnings:   66
Slice Logic Utilization:
  Number of Slice Registers:                5,372 out of  28,800   18%
    Number used as Flip Flops:              5,355
    Number used as Latch-thrus:                17
  Number of Slice LUTs:                     5,307 out of  28,800   18%
    Number used as logic:                   5,138 out of  28,800   17%
      Number using O6 output only:          4,890
      Number using O5 output only:             75
      Number using O5 and O6:                 173
    Number used as Memory:                    155 out of   7,680    2%
      Number used as Dual Port RAM:            64
        Number using O5 and O6:                64
      Number used as Shift Register:           91
        Number using O6 output only:           90
        Number using O5 output only:            1
    Number used as exclusive route-thru:       14
  Number of route-thrus:                       91
    Number using O6 output only:               85
    Number using O5 output only:                3
    Number using O5 and O6:                     3
```

图3　Magpie密码实现面积

Slices 是 FPGA 上的面积单元,LUT 是 FPGA 的查表单元,反映到芯片的实现上就是占用面积的多少. 从图 3 中可以计算出 Magpie 占用的 Slices 面积大小为:Size = 10679 slices.

图 4 是 Magpie 密码算法加密运算的工作频率,可以看出系统时钟周期是 9. 866ns,时钟频率 101. 358 MHz. 由此可以计算出的 Magpie 密码算法吞吐率为:6. 4869Gb/s.

```
Timing summary:
---------------

Timing errors: 0  Score: 0 (Setup/Max: 0, Hold: 0)

Constraints cover 396381 paths, 16 nets, and 28944 connections

Design statistics:
   Minimum period:   9.866ns (Maximum frequency: 101.358MHz)
   Maximum path delay from/to any node:   4.405ns
   Maximum net delay:   0.838ns


Analysis completed Sun Oct 13 19:35:42 2013
```

图4　Magpie密码工作频率

同样,对目前大家广为关注的轻量级密码算法 PRESENT 我们也在同样的实验环境下进行了面积与性能测试,图 5 是 PRESENT 密码算法的面积占用实验结果,图 6 是工作频率.

由图 5 和图 6 可以计算出 PRESENT 占用面积为 11824 slices;吞吐率为: 1. 8885Gb/s. 由上实验对比可

```
Design Summary:
Number of errors:      0
Number of warnings:   66
Slice Logic Utilization:
  Number of Slice Registers:                5,292 out of  28,800   18%
    Number used as Flip Flops:              5,276
    Number used as Latch-thrus:                16
  Number of Slice LUTs:                     6,532 out of  28,800   22%
    Number used as logic:                   6,363 out of  28,800   22%
      Number using O6 output only:          6,116
      Number using O5 output only:             75
      Number using O5 and O6:                 172
    Number used as Memory:                    155 out of   7,680    2%
      Number used as Dual Port RAM:            64
        Number using O5 and O6:                64
      Number used as Shift Register:           91
        Number using O6 output only:           90
        Number using O5 output only:            1
    Number used as exclusive route-thru:       14
  Number of route-thrus:                       90
    Number using O6 output only:               85
    Number using O5 output only:                2
    Number using O5 and O6:                     3
```

图5 PRESENT密码实现面积

以看出 Magpie 密码算法在面积和加密性能上有明显优势.

```
Timing summary:
---------------

Timing errors: 70  Score: 1464544 (Setup/Max: 1464544, Hold: 0)

Constraints cover 295147905179353090000 paths, 16 nets, and 37785 connections

Design statistics:
   Minimum period:  33.888ns (Maximum frequency:  29.509MHz)
   Maximum path delay from/to any node:   4.956ns
   Maximum net delay:   0.838ns


Analysis completed Fri Aug 16 16:37:35 2013
```

图6 PRESENT密码工作频率

7 结论

论文提出了一种新的基于 SPN 结构的轻量级分组密码算法 Magpie. Magpie 分组长度为 64 位,密钥长度为 96 位. 基于 SPN 结构的分组密码优点是扩散性好,缺点是加解密不一致,而我们提出的 Magpie 密码算法加解密结构完全一致,可以有效节省内存的使用和减少算法复杂度. 同时,Magpie 密码算法的加解密运算流程由输入的密钥随机决定,因此,有效地提高了密码算法自身的安全性. 下一步工作是继续对 64 位与 128 位密钥长的 Magpie 密码算法进行设计与实现,并且进一步优化密码算法结构,使其硬件实现面积更小,加密速度更快,抗攻击能力更强. 相关测试向量在文后附录部分,可供研究者使用与验证.

参考文献

[1] A Bogdanov, L R Knudsen, G Leander, et al. PRESENT: an ultra-lightweight block cipher [A]. CHES 2007[C]. Vienna, Austria, LNCS 4727, 2007. 450 – 466.

[2] H J Cheng, M H Heys, C Wang. PUFFIN: a novel compact block cipher targeted to embedded digital systems [A]. DSD 2008 [C]. Parma, Italy. IEEE Computer Society, 2008. 383 – 390.

[3] M Izadi, B Sadeghiyan, S Sadeghian, H Khanooki. MIBS: a new lightweight block cipher [A]. CANS 2009[C]. Kanazawa, Japan, LNCS 5888, 2009. 334 – 348.

[4] L R Knudsen, G Leander, A Poschmann, M J B Robshaw. PRINTcipher: a block cipher for IC-printing [A]. CHES 2010[C]. Santa Barbara, USA, LNCS 6225, 2010. 16 – 32.

[5] J Guo, T Peyrin, A Poschmann, and M J B Robshaw. The LED block cipher [A]. CHES 2011 [C]. Nara, Japan, LNCS 6917, 2011. 326 – 341.

[6] H Yap, K Khoo, A Poschmann, M Henricksen. EPCBC-a block cipher suitable for electronic product code encryption [A]. CANS 2011 [C]. Sanya China, LNCS 7092, 2011. 76 – 97.

[7] W L Wu, L Zhang. LBlock: a lightweight block cipher [A]. ANCS 2011[C]. Nerja Spain, LNCS 6715, 2011. 327 – 344.

[8] K Shibutani, T Isobe, H Hiwatari, et al. Piccolo: an ultra-lightweight blockcipher[A]. CHES 2011 [C]. Nara Japan, LNCS 6917, 2011. 342 – 357.

[9] Z Gong, S Nikova, Y W Law. KLEIN: a new family of lightweight block ciphers [J]. RFID Security and Privacy, 2012 , LNCS 7055: 1 – 18.

[10] T Suzaki, K Minematsu, S Morioka, E Kobayashi. Twine: a lightweight block cipher for multiple platforms [A]. SAC 2012[C]. Windsor, Canada, LNCS 7707. 339 – 354.

[11] J Borghoff, A Canteaut, T Guneysu, et al. PRINCE-a low-latency block cipher for pervasive computing applications [A]. ASIACRYPT 2012 [C]. Beijing, China, LNCS 7658, 2012. 208 – 225.

[12] L Keliher. Refined analysis of bounds related to linear and differentail cryptanalysis for the AES [A]. AES 2004 [C]. Bonn, Germany, LNCS 3373, 2005. 42 – 57.

[13] 彭昌勇,朱创营,黄莉,祝跃飞,王靳辉. 扩展的代数侧信道攻击及其应用[J]. 电子学报,2013,41(5):859 – 863.
Peng Chang-yong, Zhu Chuang-ying, Huang Li, et al. Extended algebraic-side channel attack and its application [J]. Acta Electronica Sinica, 2013, 41(5): 859 – 863. (in Chinese)

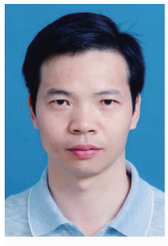
[14] T Nicolas. Courtois and J Pieprzyk. Crypt analysis of block ciphers with overdefined systems of equations [A]. ASIACRYPT 2002 [C]. Queenstown, New Zealand, LNCS 2501, 2002. 267 – 287.

附录 A: Test Vectors

Plaintext	key	Ciphertext
0123_4567_89AB_CDEF	0123_4567_89AB_CDEF_0123_4567	FA6E_88AA_D71A_45E2
FA6E_88AA_D71A_45E2	9C2B_1467_35A8_EDCF_FEDC_195D	0123_4567_89AB_CDEF
A5DE_14CF_3BB5_8740	0876_7877_CB53_381B_77E6_4B65	4CDE_AEDC_9949_6779
4CDE_AEDC_9949_6779	9B76_7B77_139B_BB48_8819_592D	A5DE_14CF_3BB5_8740

作者简介

李浪(通讯作者)　男,1971 年生于湖南衡阳,教授,博士,硕士生导师,主要研究方向为嵌入式计算与信息安全.
E-mail:lilang911@126.com



李肯立　男,1971 年生于湖南涟源,教授,博士生导师,主要研究方向为高性能计算与信息安全.

贺位位　男,1989 年生于湖南衡阳,硕士,主要研究方向为嵌入式计算与信息安全.

邹　祎　女,1983 年生于湖南衡阳,硕士,讲师,主要研究方向为嵌入式计算与信息安全.

刘波涛　男,1991 年生于湖南攸县,主要研究方向为嵌入式计算与信息安全.