GMI
使用手册

初步修订 Rev. 1.0
5/23/2016

上海兆芯集成电路有限公司

版权声明:

免责声明:

本产品非针对国防、军事、或航空等应用目的所设计。上海兆芯集成电路有限公司未授权或暗示授权任何相关的专利, 专利权与许可。上海兆芯集成电路有限公司不对本文件及本文档中描述的产品作出任何隐含或其它的保证。上海兆芯集成电路有限公司对本文档中的任何错误概不承担责任。此外，上海兆芯集成电路有限公司对使用或误用本文件信息，以及由于使用本文件可能产生的任何专利侵权概不承担任何责任。本文档中的信息和产品规格可能随时修改，恕不另行通知，且无义务通知任何人此类修改。

公司地址:

上海
Shanghai

上海兆芯集成电路有限公司
上海市浦东新区金科路 2860 号
（201203）
Tel: +86-021-6061-1988

北京
Beijing

北京兆芯电子科技有限公司
北京市海淀区中关村东路 1 号院 7 号楼
威盛中国芯大厦
(100084)
Tel: +86-021-6061-1988

修订纪录

文件版本	日期	修订叙述	编辑
1.0	5/20/16	Preliminary initial release	DA

目录

1. SM3/SM4 指令详解

GMI SM3/SM4 是兆芯通过给 CPU 增加新的指令，使 CPU 支持 SM3 以及 SM4 算法。SM4 算法的五种工作模式 ECB，CBC，CFB，OFB，CTR 模式本 CPU 也全部支持。

GMI 所有支持的指令集如下：

指令	Opcode	说明
CCS_HASH	0xF3 0x0F 0xA6 0xE8	SM3 指令
CCS_ENCRYPT	0xF3 0x0F 0xA7 0xF0	SM4 指令

1.1 *GMI SM3*

GMI SM3 基本指令概况（以 32 位系统为例）如下表所示：

指令	CCS_HASH	
Opcode	0xF3 0x0F 0xA6 0xE8	
Input Register	EAX	当 EAX=0，则执行 padding; 当 EAX=-1，则不执行 padding。
	EBX	等于 0x20，则认为 SM3 Function 被使能。
	ECX	输入 massage 的大小： 当 EAX=0，以 byte 为单位计算； 当 EAX=-1，以 block（ 64 bytes）为单位计算。
	RSI	指向输入的 massage。
	RDI	指向存放初始摘要值的内存空间。
Output Register	EAX	当 EAX=0，则执行完指令后，EAX 等于 ECX； 当 EAX=-1，则不变化。
	EBX	不变化。
	ECX	当 EAX=0，则执行完指令后，ECX 不变化； 当 EAX=-1，则 ECX=0。
	ESI	指向待执行的数据。
	EDI	不变化。最终计算出来的摘要值存放于该地址指向的内存空间。

1.2 *GMI SM4*

GMI SM4 基本指令概况（以 32 位系统为例）如下表所示：

指令	CCS_ENCRYPT	
Opcode	0xF3 0x0F 0xA7 0xF0	
Input Register	EAX	· 当 Bit[0]=0，做加密运算；当 Bit[0]=1，做解密运算。 · Bit[5:1]=10000，使能 SM4 功能。 · Bit[10:6]:SM4 模式选择 Bit 6: ECB mode Bit 7: CBC mode Bit 8: CFB mode Bit 9: OFB mode Bit 10: CTR mode · 当 Bit[11]=1，执行 MAC 操作；否则不执行。且仅针对 CBC 和 CFB 模式有效。
	EBX	指向 key。
	ECX	要被加密或解密的数据长度。单位是 128-bits 的个数。
	EDX	指向 IV。
	ESI	指向输入 massage。
	EDI	指向加密/解密结果。
Output Register	EAX	不变化。
	EBX	不变化。
	ECX	0
	ESI	指向当前待执行的数据。
	EDI	指向当前加密/解密的结果。

2. Sample Code

2.1 *GMI SM3 Sample Code*

- 32 位系统：

```
.size  gmi_sm3_oneshot,.-.L_gmi_sm3_oneshot_begin
.globl gmi_sm3_blocks
.type  gmi_sm3_blocks,@function
.align 16
gmi_sm3_blocks:
.L_gmi_sm3_blocks_begin:
        pushl %ebx
        pushl %edi
        pushl %esi
        movl  16(%esp),%edi
        movl  20(%esp),%esi
        movl  24(%esp),%ecx
        movl  %esp,%edx
        addl  $-128,%esp
        movups      (%edi),%xmm0
        andl  $-16,%esp
        movups      16(%edi),%xmm1
        movaps      %xmm0,(%esp)
        movl  %esp,%edi
        movaps      %xmm1,16(%esp)
        movl  $32,%ebx
        movl  $-1,%eax
.byte  0xf3,0x0f,0xa6,0xe8
        movaps      (%esp),%xmm0
        movaps      16(%esp),%xmm1
        movl  %edx,%esp
        movl  16(%esp),%edi
        movups      %xmm0,(%edi)
        movups      %xmm1,16(%edi)
        popl  %esi
        popl  %edi
        popl  %ebx
        ret
.size  gmi_sm3_blocks,.-.L_gmi_sm3_blocks_begin
```

- 64 位系统：

```
.globl gmi_sm3_blocks
.type gmi_sm3_blocks,@function
.align 16
gmi_sm3_blocks:
        movq %rbx,%r11
        movq %rdx,%rcx
        movq %rdi,%rdx
        movups      (%rdi),%xmm0
        subq  $128+8,%rsp
        movups      16(%rdi),%xmm1
        movaps      %xmm0,(%rsp)
        movq %rsp,%rdi
        movaps      %xmm1,16(%rsp)
        movq $32,%rbx
        movq $-1,%rax
.byte 0xf3,0x0f,0xa6,0xe8
        movaps      (%rsp),%xmm0
        movaps      16(%rsp),%xmm1
        addq $128+8,%rsp
        movups      %xmm0,(%rdx)
        movups      %xmm1,16(%rdx)
        movq %r11,%rbx
        .byte 0xf3,0xc3
.size  gmi_sm3_blocks,.-gmi_sm3_blocks
```

2.2 GMI SM4 Sample Code

- 32 位系统：

```
.globl gmi_gx6_sm4_encrypt
.type gmi_gx6_sm4_encrypt,@function
.align 16
gmi_gx6_sm4_encrypt:
.L_gmi_gx6_sm4_encrypt_begin:
        pushl %ebx
        pushl %edi
        pushl %esi
        movl 16(%esp),%edi
        movl 20(%esp),%esi
        movl 24(%esp),%edx
        movl 28(%esp),%ecx
        leal  32(%edx),%ebx
        shrl  $4,%ecx
        movl 16(%edx),%eax
.byte 0xf3,0x0f,0xa7,0xf0
        popl  %esi
        popl  %edi
        popl  %ebx
        ret
.size  gmi_gx6_sm4_encrypt,.-.L_gmi_gx6_sm4_encrypt_begin
```

- 64 位系统：

```
.globl gmi_gx6_sm4_encrypt
.type gmi_gx6_sm4_encrypt,@function
.align 16
gmi_gx6_sm4_encrypt:
        pushq       %rbp
        pushq       %rbx
        pushq       %rdi
        pushq       %rsi
        leaq  32(%rdx),%rbx
        shrq  $4,%rcx
        movq 16(%rdx),%rax
.byte 0xf3,0x0f,0xa7,0xf0
        popq %rsi
        popq %rdi
        popq %rbx
        popq %rbp
        .byte 0xf3,0xc3
.size  gmi_gx6_sm4_encrypt,.-gmi_gx6_sm4_encrypt
```