实验目标：

实现一个能够在ARM系统上运行的AES加密（解密同理）

实验过程：

首先需要一个能够运行ARM指令的环境，当前环境为X86架构，无法运行ARM指令集。考虑使用ARM架构虚拟机。选择ARMv8Ubuntu，安装后弹出设备架构为X86该虚拟机无法运行。因此考虑使用QEMU或GEM5等仿真器进行模拟。、

选择ARM版本为ARMv8。

查阅ARMv8手册可以找到密码学拓展指令集AES部分。主要有以下4条指令：AESE、AESD、AESMC、AESIMC。分别为AES单轮加密(实际只包含AddRoundKey，ShiftRow ，SubByte)、AES单轮解密、MixColumn、Mixcolumn逆。

**AESE**

AESE <Vd>.16B, <Vn>.16B

功能：

AArch64.CheckFPAdvSIMDEnabled();

bits(128) operand1 = V[d, 128];

bits(128) operand2 = V[n, 128];

bits(128) result;

result = operand1 EOR operand2;

result = AESSubBytes(AESShiftRows(result));

V[d, 128] = result;

**AESD**

AESDC<Vd>.16B, <Vn>.16B

操作：

AArch64.CheckFPAdvSIMDEnabled();

bits(128) operand1 = V[d, 128];

bits(128) operand2 = V[n, 128];

bits(128) result;

result = operand1 EOR operand2;

result = AESInvSubBytes(AESInvShiftRows(result));

V[d, 128] = result;

**AESMC**

AESMC <Vd>.16B, <Vn>.16B

操作：

AArch64.CheckFPAdvSIMDEnabled();

bits(128) operand = V[n, 128];

bits(128) result;

result = AESMixColumns(operand);

V[d, 128] = result;

**AESIMC**

AESIMC <Vd>.16B, <Vn>.16B

操作：

AArch64.CheckFPAdvSIMDEnabled();

bits(128) operand = V[n, 128];

bits(128) result;

result = AESInvMixColumns(operand);

V[d, 128] = result;

参考代码：

void aes128(const uint8\_t in[16],uint8\_t ou[16],const uint32\_t rk[44]) {

uint8x16\_t block = vld1q\_u8(in);

uint8\_t\* p8 = (uint8\_t\*)rk;

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 0)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 1)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 2)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 3)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 4)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 5)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 6)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 7)));

block = vaesmcq\_u8(vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 8)));

block = vaeseq\_u8(block, vld1q\_u8(p8 + 16 \* 9));

block = veorq\_u8(block, vld1q\_u8(p8 + 16 \* 10));

vst1q\_u8(ou, block);

}

首先观察到轮加密时先传入了轮密钥，这意味着先进行了轮密钥加。根据AES原理，ShiftRow和MixColumn都会破坏AddRoundKey的作用（反之亦然），因此可以断定代码结构并非传统AES执行结构（但是是传统AES结构）。

结合ARM拓展指令集分析得知，此代码将轮过程改为

1. AddRoundKey
2. ShiftRow
3. SubByte
4. MixColumn

将初始轮密钥加并入第一轮并将末尾轮密钥加移入下一轮。最后一轮恰好是将列混合改为轮密钥加。

由是得知mcq是MixColumn,eq是ShiftRow+SubBytes，可以定义：

uint8x16\_t vaeseq\_u8(uint8x16\_t block, uint8x16\_t key) {

/\*

\* AESE <block>.16B, <key>.16B

\*

\* return block;

\*/

}

uint8x16\_t vaesmcq\_u8(uint8x16\_t p) {

/\*

\* uint8x16\_t Vd;

\*

\* AESMC <Vd>.16B, <p>.16B

\*

\* return Vd;

\*/

}