ARMv8상에서 Classic McEliece의 Multiplication and Inversion operation 구현

https://youtu.be/g43mKHFng94





Classic McEliece

- NIST PQC 4라운드 후보군 중 유일한 코드기반 암호
- Key generation, Encapsulation, Decapsulation 3단계 수행
- Extended binary finite-filed F_{2^m} 상의 Multiplication 과 Inversion 연산
 - Public keygen : inverse + multiplication on F_{2^m}
 - Secret keygen : inverse + multiplication on F_{2^m}
 - Encap : X
 - Decap : inverse + multiplication on F_{2^m}

$$\mathbf{F}_{2^{12}} = \mathbb{F}_2[x]/(x^{12} + x^3 + 1).$$

$$F_{2^{13}} = \mathbb{F}_2[x] / (x^{13} + x^4 + x^3 + x + 1).$$

Algorithm	m	n	t	level	Public key	Secret key	Ciphertext
Mceliece 348864	12	3488	64	1	261,120	6,492	128
Mceliece 460896	13	4608	86	3	524,160	13,608	188
Mceliece 6688128	13	6688	128	5	1,044,992	13,932	240
Mceliece 6960119	13	6960	119	5	1,047,319	13,948	226
Mceliece 8192128	13	8192	128	5	1,357,824	14,120	240

• Secret key gen의 일부

```
/* input: f, element in GF((2^m)^t) */
/* output: out, minimal polynomial of f */
/* return: 0 for success and -1 for failure */
int PQCLEAN_MCELIECE348864_CLEAN_genpoly_gen(gf *out, gf *f) {
    int i, j, k, c;
    gf mat[ SYS_T + 1 ][ SYS_T ];
    gf mask, inv, t;
    // fill matrix
    mat[0][0] = 1;
    for (i = 1; i < SYS_T; i++) {
        mat[0][i] = 0;
    for (i = 0; i < SYS T; i++) {
        mat[1][i] = f[i];
    }
    for (j = 2; j <= SYS_T; j++) {</pre>
        PQCLEAN_MCELIECE348864_CLEAN_GF_mul(mat[j], mat[j - 1], f);
    // gaussian
    for (j = 0; j < SYS_T; j++) {</pre>
        for (k = j + 1; k < SYS T; k++) {
            mask = PQCLEAN_MCELIECE348864_CLEAN_gf_iszero(mat[ j ][ j ]);
            for (c = j; c < SYS_T + 1; c++) {
                mat[ c ][ j ] ^= mat[ c ][ k ] & mask;
```

```
int PQCLEAN_MCELIECE348864_CLEAN_genpoly_gen(gf *out, gf *f) {
    int i, j, k, c;
    gf mat[(SYS_T + 1)*SYS_T];
                                          //4160
    gf mask, inv, t;
    mat[0] = 1;
    for (i = 1; i < SYS T; i++) { //1~63}
        mat[0+i] = 0;
    for (i = 0; i < SYS_T; i++) { //64~127}
        mat[SYS_T+i] = f[i];
  // for (j = 2; j \le SYS_T-1; j++) {
    for (j = 2; j <= SYS_T; j++) {
         PQCLEAN MCELIECE348864 CLEAN GF mul(&mat[j*SYS T], &mat[(j - 1)*SYS T], f);
        // GF_mul(&mat[j*SYS_T], &mat[(j - 1)*SYS_T], f);
    }
    // gaussian
    for (j = 0; j < SYS_T; j++) {
       for (k = j + 1; k < SYS_T; k++) {
            mask = PQCLEAN MCELIECE348864 CLEAN gf iszero(mat[ (j*SYS T)+j ]);
           //0, 65, 130, 195 , 260 ..4030 ->65씩 증가
            for (c = j; c < SYS_T + 1; c++) {
                mat[(SYS_T * c)+j]^= mat[(SYS_T * c)+k] \& mask;
```

```
/* input: in0, in1 in GF((2^m)^t)*/
/* m : 12, t : 64*/
/* output: out = in0*in1 */
/*secretkey sk_gen에서만 필요한 함수*/
void PQCLEAN_MCELIECE348864_CLEAN_GF_mul(gf *out, const gf *in0, const gf *in1) {
   int i, j;
   gf prod[ SYS_T * 2 - 1 ];
    for (i = 0; i < SYS_T * 2 - 1; i++) {
       prod[i] = 0;
   for (i = 0; i < SYS T; i++) {
       for (j = 0; j < SYS_T; j++) {
           prod[i + j] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(in0[i], in1[j]);
   11
   for (i = (SYS_T - 1) * 2; i >= SYS_T; i--) {
       prod[i - SYS_T + 9] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 877);
       //877 -> 0x36D
       prod[i - SYS_T + 7] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 2888);
       //2888 ->0xB48
       prod[i - SYS_T + 5] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 1781);
       //0x6F5
       prod[i - SYS_T + 0] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 373);
       //0x175
   for (i = 0; i < SYS_T; i++) {
       out[i] = prod[i];
```

GF_mul:		loop2:	
_GF_mul:		add	x0, x0, #124
mov	w3, #1	ldrh	w23, [x0]
mov	w8, #64	mov	w10, w23
mov	w9, #64 //index i	mov	w20, #0x36D
mov	w12, #63	gf_mul	•
mov	w11, #128	mov	w24, w13
//첫번째 매개변수(d	out) 초기화	1 -11-	
loop0:		ldrh	w23, [x0]
mov	w23, #0	mov	w10, w23
strh	w23, [x0], #2	mov	w20, #0xB48
		gf_mul	
add	w11, w11, #-1	mov	w25, w13
cbnz	w11, loop0		
		ldrh	w23, [x0]
///////////////////////////////////////	////	mov	w10, w23
add	x0, x0, #-256	mov	w20, #0x6F5
ldrh	w21, [x1]	gf_mul	
		mov	w26, w13
//////////		ldrh	w23, [x0]
loop:		mov	w10, w23
ldrh	w22, [x2]	mov	w10, w23 w20, #0x175
ldrh	w23, [x0]		WZ0, #0X1/5
		gf_mul	07 40
mov	w10, w21	mov	w27, w13
mov	w20, w22		
gf_mul	•	add	x0, x0, #-110
eor	w23, w23, w13		
add	x2, x2, #2	ldrh	w23, [x0]
strh	w23, [x0],#2	eor	w23, w23, w24
		strh	w23, [x0], #-4
add	w8, w8, #-1		
cbnz	w8, loop	ldrh	w23, [x0]
cbz	w8, loop1	eor	w23, w23, w25
		strh	w23, [x0], #-4
loop1:		ldrh	w23, [x0]
add	x1, x1, #2	eor	w23, w23, w26
ldrh	w21, [x1]	strh	w23, [x0], #-10
add	x0, x0, #-126	3(11)	W25, [X0], #-16
add	x2, x2, #-128	ldrh	maa [va]
mov	w8, #64	ldrh	w23, [x0]
add	w9, w9, #-1	eor	w23, w23, w27
cbnz	w9, loop	strh	w23, [x0],#2
		add	w12, w12, #-1
cbz	w9, loop2	cbnz	w12, loop2

```
/* input: in0, in1 in GF((2^m)^t)*/
/* m : 12, t : 64*/
/* output: out = in0*in1 */
/*secretkey sk_gen에서만 필요한 함수*/
void PQCLEAN_MCELIECE348864_CLEAN_GF_mul(gf *out, const gf *in0, const gf *in1) {
    int i, j;
    gf prod[ SYS_T * 2 - 1 ];
    for (i = 0; i < SYS_T * 2 - 1; i++) {
        prod[i] = 0;
    for (i = 0; i < SYS_T; i++) {
        for (j = 0; j < SYS_T; j++) {
            prod[i + j] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(in0[i], in1[j]);
    11
    for (i = (SYS_T - 1) * 2; i >= SYS_T; i--) {
       prod[i - SYS_T + 9] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 877);
       //877 -> 0x36D
       prod[i - SYS_T + 7] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 2888);
       //2888 ->0xB48
        prod[i - SYS_T + 5] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 1781);
        //0x6F5
        prod[i - SYS_T + 0] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 373);
        //0x175
    for (i = 0; i < SYS_T; i++) {
        out[i] = prod[i];
```

GF_mul:		loop2:	
_GF_mul:		add	x0, x0, #124
_01 _11141.		auu	X0, X0, #124
mov	w3, #1	ldrh	w23, [x0]
mov	w8, #64	mov	w10, w23
mov	w9, #64 //index i		
mov	w12, #63	mov	w20, #0x36D
mov	w11, #128	gf_mul	
1110 V	WIII # # 120	mov	w24, w13
//첫번째 매개변	수(out) 초기화		
loop0:		ldrh	w23, [x0]
mov	w23, #0	mov	w10, w23
strh	w23, [x0], #2	mov	w20, #0xB48
0 0 0 1 11	"207 [X0]7 "2	gf_mul	
add	w11, w11, #-1	mov	w25, w13
cbnz	w11, loop0		
OBITE	W117 100p0	ldrh	w23, [x0]
///////////////////////////////////////	///////	mov	w10, w23
add	x0, x0, #-256	mov	w20, #0x6F5
ldrh	w21, [x1]	gf_mul	11207 1100010
10111	#21/ [A1]	mov	w26, w13
		IIIO V	W20, W13
///////////////////////////////////////		ldrh	w23, [x0]
loop:		mov	w10, w23
ldrh	w22, [x2]		w20, #0x175
ldrh	w23, [x0]	MOV	WZ0, #0X1/5
		gf_mul	07 40
mov	w10, w21	mov	w27, w13
mov	w20, w22		
gf_mul	·	add	x0, x0, #-110
eor	w23, w23, w13		
add	x2, x2, #2	ldrh	w23, [x0]
strh	w23, [x0],#2	eor	w23, w23, w24
		strh	w23, [x0], #-4
add	w8, w8, #-1		
cbnz	w8, loop	ldrh	w23, [x0]
cbz	w8, loop1	eor	w23, w23, w25
		strh	w23, [x0], #-4
			,,
loop1:		ldrh	w23, [x0]
add	x1, x1, #2	eor	w23, w23, w26
ldrh	w21, [x1]	strh	w23, [x0], #-10
add	x0, x0, #-126	0 1211	11207 [203] 11 20
add	x2, x2, #-128	ldrh	w23, [x0]
mov	w8, #64		w23, [x0] w23, w23, w27
add	w9, w9, #-1	eor	
cbnz	w9, loop	strh	w23, [x0],#2
cbz	w9, loop2	add	w12, w12, #-1
		cbnz	w12, loop2
///////			

Algorithm	m	n	t	level	Public key	Secret key	Ciphertext
Mceliece 348864	12	3488	64	1	261,120	6,492	128
Mceliece 460896	13	4608	86	3	524,160	13,608	188
Mceliece 6688128	13	6688	128	5	1,044,992	13,932	240
Mceliece 6960119	13	6960	119	5	1,047,319	13,948	226
Mceliece 8192128	13	8192	128	5	1,357,824	14,120	240
Mceliece 8192128	13	8192	128	5	1,357,824	14,120	240

```
/* input: in0, in1 in GF((2^m)^t)*/
/* m : 12, t : 64*/
/* output: out = in0*in1 */
/*secretkey sk gen에서만 필요한 함수*/
void PQCLEAN_MCELIECE348864_CLEAN_GF_mul(gf *out, const gf *in0, const gf *in1) {
   int i, j;
    gf prod[ SYS_T * 2 - 1 ];
    for (i = 0; i < SYS_T * 2 - 1; i++) {
        prod[i] = 0;
    for (i = 0; i < SYS_T; i++) {
        for (j = 0; j < SYS_T; j++) {
            prod[i + j] ^= PQCLEAN MCELIECE348864 CLEAN gf mul(in0[i], in1[j]);
    11
    for (i = (SYS T - 1) * 2; i >= SYS T; i--) {
        prod[i - SYS_T + 9] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 877);
       //877 -> 0x36D
        prod[i - SYS_T + 7] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 2888);
        prod[i - SYS_T + 5] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 1781);
        prod[i - SYS_T + 0] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 373);
        //0x175
    for (i = 0; i < SYS_T; i++) {
        out[i] = prod[i];
```

```
GF mul:
                                      loop2:
_GF_mul:
                                          add
                                                      x0, x0, #124
               w3, #1
                                          ldrh
                                                      w23, [x0]
               w8, #64
                                           mov
                                                      w10, w23
               w9, #64 //index i
                                                      w20, #0x36D
                                           mov
               w12, #63
   mov
                                          gf_mul
               w11, #128
                                           mov
                                                      w24, w13
//첫번째 매개변수(out) 초기화
                                                      w23, [x0]
                                          ldrh
loop0:
                                           mov
                                                      w10, w23
               w23, #0
                                                      w20, #0xB48
                                           mov
               w23, [x0], #2
   strh
                                          gf_mul
                                          mov
                                                      w25, w13
               w11, w11, #-1
   add
               w11, loop0
   cbnz
                                          ldrh
                                                      w23, [x0]
w10, w23
                                           mov
                                                      w20, #0x6F5
   add
               x0, x0, #-256
                                           mov
               w21, [x1]
   ldrh
                                          gf_mul
                                           mov
                                                      w26, w13
ldrh
                                                      w23, [x0]
loop:
                                                      w10, w23
                                           mov
   ldrh
               w22, [x2]
                                           mov
                                                      w20, #0x175
   ldrh
               w23, [x0]
                                          gf_mul
                                                      w27, w13
               w10, w21
   mov
   mov
               w20, w22
                                           add
                                                      x0, x0, #-110
   gf_mul
               w23, w23, w13
   eor
                                          ldrh
                                                      w23, [x0]
               x2, x2, #2
   add
                                           eor
                                                      w23, w23, w24
               w23, [x0],#2
   strh
                                                      w23, [x0], #-4
                                          strh
               w8, w8, #-1
   add
                                          ldrh
                                                      w23, [x0]
   cbnz
               w8, loop
               w8, loop1
                                                      w23, w23, w25
   cbz
                                          eor
                                          strh
                                                      w23, [x0], #-4
loop1:
                                          ldrh
                                                      w23, [x0]
   add
               x1, x1, #2
                                                      w23, w23, w26
                                           eor
               w21, [x1]
                                          strh
                                                      w23, [x0], #-10
               x0, x0, #-126
               x2, x2, #-128
                                          ldrh
                                                      w23, [x0]
               w8, #64
                                                      w23, w23, w27
                                          eor
   add
               w9, w9, #-1
                                           strh
                                                      w23, [x0],#2
   cbnz
               w9, loop
                                                      w12, w12, #-1
                                           add
               w9, loop2
                                          cbnz
                                                      w12, loop2
```

////////

```
/* input: in0, in1 in GF((2^m)^t)*/
/* m : 12, t : 64*/
/* output: out = in0*in1 */
/*secretkey sk_gen에서만 필요한 함수*/
void PQCLEAN_MCELIECE348864_CLEAN_GF_mul(gf *out, const gf *in0, const gf *in1) {
   int i, j;
   gf prod[ SYS_T * 2 - 1 ];
   for (i = 0; i < SYS_T * 2 - 1; i++) {
       prod[i] = 0;
   for (i = 0; i < SYS_T; i++) {</pre>
       for (j = 0; j < SYS_T; j++) {
           prod[i + j] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(in0[i], in1[j]);
   for (i = (SYS_T - 1) * 2; i >= SYS_T; i--) {
       prod[i - SYS_T + 9] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 877);
       //877 -> 0x36D
       prod[i - SYS_T + 7] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 2888);
       //2888 ->0xB48
       prod[i - SYS_T + 5] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 1781);
       //0x6F5
       prod[i - SYS_T + 0] ^= PQCLEAN_MCELIECE348864_CLEAN_gf_mul(prod[i], (gf) 373);
        //0x175
   for (i = 0; i < SYS_T; i++) {
       out[i] = prod[i];
```

```
gf PQCLEAN_MCELIECE348864_CLEAN_gf_mul(gf in0, gf in1) {
    int i;
    uint32_t tmp;
    uint32_t t0;
    uint32_t t1;
    uint32_t t;
    t0 = in0;
    t1 = in1;
    tmp = t0 * (t1 & 1);
    for (i = 1; i < GFBITS; i++) {</pre>
        tmp ^{-} (t0 * (t1 & (1 << i)));
    t = tmp & 0x7FC000;
    tmp ^= t >> 9;
    tmp ^{=} t >> 12;
    t = tmp & 0x3000;
    tmp ^= t >> 9;
    tmp ^= t >> 12;
    return tmp & ((1 <k GFBITS) - 1);
```

Algorithm	ı	m		n	t	level	Public key	Secret key	Ciphertext
Mceliece 348864	Г	12		3488	64	1	261,120	6,492	128
Mceliece 460896	Г	13		4608	86	3	524,160	13,608	188
Mceliece 6688128	Г	13		6688	128	5	1,044,992	13,932	240
Mceliece 6960119	Г	13	П	6960	119	5	1,047,319	13,948	226
Mceliece 8192128	Г	13		8192	128	5	1,357,824	14,120	240

.macro gf_mul and	1 /	00	44
and mul		w20, w10,	
IIIUI	WIS,	WIO,	W14
//gfbit_mul			0 1 1 1 114
and			w3, lsl #1
mul		w10,	
eor	w13,	w13,	W14
//gfbit_mul	2		
and	w14,	w20,	w3, lsl #2
mul	w14,	w10,	w14
eor	w13,	w13,	w14
//gfbit_mul			
and			w3, lsl #3
mul		w10,	
eor	w13,	w13,	w14
		:	
//gfbit_mul	11		
and	w14,	w20,	w3, lsl #11
mul	w14,	w10,	w14
eor	w13,	w13,	w14
		4.0	"0 7F0000
and			#0x7FC000
eor			w16, lsr #9
eor	W13,	W13,	w16, lsr #12
and	w16,	w13,	#0x3000
eor			w16, lsr #9
eor			w16, lsr #12
		,	•
lsl	w16,	w3, a	#12
sub		w16,	
and	w13,	w13,	w16
.endm			

loop2:	
add	x0, x0, #124
ldrh	w23, [x0] w10, w23
mov	w10, w23
mov	w20, #0x36D
gf_mul	
mov	w24, w13
ldrh	w23, [x0]
mov	w10, w23 w20, #0×B48
mov	w20, #0xB48
gf_mul	
mov	w25, w13
ldrh	w23, [x0] w10, w23
mov	w10, w23
mov	w20, #0x6F5
gf_mul	
mov	w26, w13
ldrh	w23, [x0]
mov	w10, w23
mov	w10, w23 w20, #0x175
gf_mul	
mov	w27, w13
add	x0, x0, #-110
ldrh	w23, [x0] w23, w23, w24
eor	w23, w23, w24
strh	w23, [x0], #-4
ldrh	w23, [x0]
eor	w23, w23, w25
strh	w23, w23, w25 w23, [x0], #-4
ldrh	w23, [x0]
eor	w23, w23, w26
strh	w23, [x0], #-10
ldrh	w23, [x0] w23, w23, w27
eor	w23, w23, w27
strh	w23, [x0],#2
add	w12, w12, #-1
cbnz	w12, loop2

Inversion

- Inversion 연산
 - Squaring 연산과 Multiplication 연산

```
gf PQCLEAN_MCELIECE348864_CLEAN_gf_inv(gf in) {
   gf tmp_11;
                                                         /* input: field element in */
   gf tmp_1111;
                                                         /* return: in^2 */
                                                         static inline qf qf sq(qf in) {
   gf out = in;
   out = gf_sq(out);
   tmp_11 = PQCLEAN_MCELIECE348864_CLEAN_gf_mul(out, in); // 11
   out = af sq(tmp 11);
   out = gf_sq(out);
   tmp_1111 = PQCLEAN_MCELIECE348864_CLEAN_gf_mul(out, tmp_11); // 1111
   out = gf_sq(tmp_1111);
   out = gf_sq(out);
   out = gf_sq(out);
   out = gf_sq(out);
   out = PQCLEAN MCELIECE348864 CLEAN gf mul(out, tmp 1111); // 11111111
   out = gf_sq(out);
   out = gf_sq(out);
   out = PQCLEAN_MCELIECE348864_CLEAN_gf_mul(out, tmp_11); // 1111111111
   out = gf_sq(out);
   out = PQCLEAN_MCELIECE348864_CLEAN_gf_mul(out, in); // 11111111111
    return gf_sq(out); // 111111111110
```

```
w0 : out
                                                                      w1 : in
                                                                                   //gf_sq의 return에서 필요
                                                                      w3 : 1
                                                                       w10 : gf_inv의 out (첫번째 매개변수)
                                                                      w11 : gf_sq의 temp
PQCLEAN_MCELIECE348864_CLEAN_gf_inv:
                                                                      w15 : gf_sq의 t
_PQCLEAN_MCELIECE348864_CLEAN_gf_inv:
                                                                      w16 : qf mul의 tmp
            w3, #1
                                                                      w20 : qf mul의 두번째 매개변수
            w2, w0
                                    //gf in
    mov
                                                                      w21 : gf_inv의 tmp_11
                                    // out = gf_sq_single(out);
                                                                      w22 : gf_inv의tmp_1111
    mov
            w10, w0
                                                                      */
    gf_sq
    mov
            w0, w10
                               // tmp 11 = PQCLEAN MCELIECE348864 CLEAN gf mul(out, in);
            w10, w0
                                                                  .macro gf_sq
            w20, w2
    mov
                                                                                  w11, w10, w10, lsl #8
    gf_mul
                                                                                  w10, w11, #0x00FF00FF
                                                                      and
            w21, w13
                                                                      orr
                                                                                  w11, w10, w10, lsl #4
                                                                      and
                                                                                  w10, w11, #0x0F0F0F0F
                           // out = gf_sq_single(tmp_11)
                                                                      orr
                                                                                  w11, w10, w10, lsl #2
            w10, w21
    mov
                                                                      and
                                                                                  w10, w11, #0x33333333
    gf_sq
                                                                                  w11, w10, w10, lsl #1
                                                                      orr
            w0, w10
    mov
                                                                                  w10, w11, #0x55555555
                               // out = gf_sq_single(out);
                                                                      and
                                                                                  w15, w10, #0x7FC000
    mov
            w10, w0
                                                                      eor
                                                                                  w10, w10, w15, lsr #9
    gf_sq
                                                                                  w10, w10, w15, lsr #12
                                                                      eor
            w0, w10
                                                                                  w15, w10, #0x3000
                               // tmp_1111 = PQCLEAN_MCELIECE34880
                                                                                  w10, w10, w15, lsr #9
                                                                      eor
            w10, w0
                                                                                  w10, w10, w15, lsr #12
    mov
            w20, w21
    gf_mul
                                                                     lsl
                                                                                  w15, w3, #12
            w22, w13
                                                                                  w15, w15, #1
                                                                      sub
                                                                      and
                                                                                  w10, w10, w15
```

.endm

Q&A