#1

Key switching:

$$e + \underset{}{\overset{\boxed{ct_0 + ct_1 * s_{fr}}}{\overset{\parallel}{pt}}} \;\overset{s_{fr}}{\text{———}}\; ct \in R_Q^2$$

$$P \cdot s_{fr}\vec{g} + \vec{e} \;\overset{s_{to}}{\text{———}}\; \overrightarrow{swk} \in \left(R_{QP}^2\right)^{DNUM}$$

$$\vec{g} \in \mathbb{Z}_{QP}^{DNUM}, \quad \vec{g}^{-1}(ct_1) \in R_{QP}^{DNUM}$$

$$\text{s.t. } \vec{g} \circ \vec{g}^{-1}(ct_1) = ct_1 \in R_Q$$

$$Ps_{fr}\vec{g} * \vec{g}^{-1}(ct_1) + \vec{e} * \vec{g}^{-1}(ct_1) \;\overset{s_{to}}{\text{———}}\; \vec{g}^{-1}(ct_1) * \overrightarrow{swk} \in \left(R_{QP}^2\right)^{DNUM}$$

$$\Big| \qquad\qquad \Big\downarrow \oplus$$

$$Ps_{fr} * ct_1 + \vec{e} \circ \overrightarrow{g^{-1}(ct_1)} \;\overset{s_{to}}{\text{———}}\; \in R_{QP}^2$$

$$\Big| \qquad\qquad \Big| \; RS$$

$$s_{fr} * ct_1 + \text{small error} \;\overset{s_{to}}{\text{———}}\; \tilde{ct} \in R_Q^2$$

$$\Big| \qquad\qquad \Big|$$

$$pt + \text{small error} \;\overset{s_{to}}{\text{———}}\; \boxed{[ct_0, 0] + \tilde{ct}} \in R_Q^2$$

$$\underset{out}{\parallel}$$

```
template<int N, int L, int DNUM, int K>
void ks( const uint64_t q[L],
         const uint64_t p[K],
         const uint64_t ŝwk[DNUM][2][DNUM*K+K][N],   // full level
         const uint64_t ĉt[2][L][N],
               uint64_t ôut[2][L][N]){

    uint64_t a[L][N];
    for(int i=0; i<L; i++)
    for(int j=0; j<N; j++) a[i][j]= ĉt[1][i][j];      // a=ct_1
    intt<N,L>(q,a);
```

#2

```
uint64_t  ginva [DNUM][L+K][N];
gadget_ginva <N,L,DNUM,K> (q, P, a, ginva);
```

$\vec{g}^{-1}(a) \in R_{QP}^{DNUM}$

```
uint64_t qp [L+K];
for(int i=0; i<L; i++) qp[i] = q[i];
for(int i=0; i<K; i++) qp[L+i] = p[i];
```

$qp = [q_0 \cdots q_{L-1} p_0 \cdots p_{K-1}]$

```
uint64_t  sum[2][L+K][N];
for(int i=0; i<L+K; i++)
for(int j=0; j<N ; j++) { sum[0][i][j]=0;
                          sum[1][i][j]=0; }
```

$sum = \sum_d \vec{g}^{-1}(a)[d] * swk[d]$ in R_{QP}

```
for(int d=0; d<DNUM; d++){
    ntt<N, L+K> (qp, ginva[d]);
    for( int i=0; i<L; i++)
    for( int j=0; j<N; j++){
        sum[0][i][j] = (sum[0][i][j] + mul_mod(ginva[d][i][j],
                                      ŝwk[d][0][i][j], q[i]) % q[i];

        sum[1][i][j] = (sum[1][i][j] + mul_mod(ginva[d][i][j],
                                      ŝwk[d][1][i][j], q[i]) % q[i];

    }
    for(int i=0; i<K; i++)
    for(int j=0; j<N; j++){
        sum[0][L+i][j] = (sum[0][L+i][j] + mul_mod(ginva[d][L+i][j],
                                      ŝwk[d][0][DNUM*K+i][j], p[i]) % p[i];

        sum[1][L+i][j] = (sum[1][L+i][j] + mul_mod(ginva[d][L+i][j],
                                      ŝwk[d][1][DNUM*K+i][j], p[i]) % p[i];

    }
}
RS_hat<N, L+K, L> ( qp, sum, ôut );
for(int i=0; i<L; i++)
for(int j=0; j<N; j++) ôut[0][i][j] = (ôut[0][i][j] + t̂[0][i][j]) % q[i];

}
```

modUp operation

$$QP = \boxed{q_0 \cdots q_{L-1}}\boxed{p_0 \cdots p_{K-1}}$$

$$a = \boxed{[a]_Q \quad \underbrace{/////////}_{\text{extended}}}$$

$$a = \sum_{j=0}^{L-1} a[j] \frac{Q}{q[j]} \left(\frac{Q}{q[j]}\right)^{-1}_{q[j]}$$

```
template<int N, int LplusK>
void modUp( const uint64_t qp[LplusK],  int L,
                uint64_t a[LplusK][N]){
    const uint64_t* q = qp;      int K=LplusK-L;
    const uint64_t* p = qp+L;
```

$table1[j][k] = mod\left(\frac{Q}{q[j]}, p[k]\right)$

```
    uint64_t table1[LplusK][LplusK];
    for(int j=0; j<L; j++)
    for (int k=0; k<K; k++){
        table1[j][k]=1;
        for(int i=0; i<L; i++)
            if(i!=j) table1[j][k] = mul_mod(table1[j][k],
                                  q[i] % p[k], p[k]);
```

$table2[j] = inv\_mod\left(\frac{Q}{q[j]}, q[j]\right)$

```
    uint64_t table2[LplusK];
    for(int j=0; j<L; j++){
        table2[j]= 1;
        for(int i=0; i<L; i++)
            if(i!=j) table2[j] = mul_mod( table2[j],
                         inv_mod(q[i]%q[j], q[j]), q[j]);
```

$table3[k] = mod(Q, p[k]);$

```
    uint64_t table3[LplusK];
    for( int k=0; k<K; k++){
        table3[k]=1;
        for(int j=0; j<L; j++)
            table3[k]= mul_mod(table3[k],
                          q[j] % p[k], p[k]);
    }
```

```
for(int i=0; i<N; i++){
    uint64_t b[Lplusk];   int count=0;

    for(int j=0; j<L; j++){
        b[j]= mul_mod(a[j][i],
                      table2[j], q[j]);
        if(2*b[j] >= q[j]) count++;
    }

    for(int k=0; k<K; k++){
        a[L+k][i]=0;

        for(int j=0; j<L; j++)
            a[L+k][i]= (a[L+k][i] +
                        mul_mod(b[j]% p[k], table1[j][k], p[k]) )% p[k];

        if(count>0)
            a[L+k][i]= (a[L+k][i]+ mul_mod(table3[k], p[k]-count, p[k]))
                                                   % p[k];

    }
}
```

$b[j] = \underbrace{a[j] \cdot \left(\frac{Q}{q[j]}\right)^{-1}}_{\text{in } q[j]}$

count++ if $b[j] \geq \frac{q[j]}{2}$

$a[L+k]$
$= \mathrm{mod}\left(\sum_j b[j]\,\frac{Q}{q[j]},\ p[k]\right)$
$- \text{count}\cdot Q$

Switch key is generated at the full level and reused at every level

Full level ($L = DNUM \times K$) e.g. $L=9$

$$Q_9 = \underbrace{q_0 q_1 q_2}_{D_0}\ \underbrace{q_3 q_4 q_5}_{D_1}\ \underbrace{q_6 q_7 q_8}_{D_2}, \quad P = \underbrace{p_0 p_1 p_2}$$

$$S_{fr}, S_{to} \in \mathbb{Z}[x]/_{x^N+1} \implies swk \in (R_{Q_9 \cdot P})^{2\times DNUM}$$

$$\big\downarrow \text{modDown to } Q_5 \cdot P$$

At level 5,

$$Q_5 = \underbrace{q_0 q_1 q_2}_{D_0}\ \underbrace{q_3 q_4}_{D_1}\ \underbrace{1}_{D_2}, \quad P = p_0 p_1 p_2$$

$$\vec{g} = [g_0, g_1, g_2] \in \mathbb{Z}_{Q_5 P}^{DNUM},$$

$$P\vec{g} \in \mathbb{Z}_{Q_5 P}^{DNUM}$$

$$\mathbb{Z}_{Q_5 P} \cong \mathbb{Z}_{q_0} \times \cdots \times \mathbb{Z}_{q_4} \times \mathbb{Z}_{p_0} \times \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2}$$

$$P g_0 = [[P]_{q_0}, [P]_{q_1}, [P]_{q_2}, 0, 0, 0, 0, 0]$$

$$P g_1 = [0, 0, 0, [P]_{q_0}, [P]_{q_1}, 0, 0, 0]$$

$$P g_2 = [0, 0, 0, 0, 0, 0, 0, 0]$$

$$\vec{g}^{-1} : \mathbb{Z}_{Q_5} \to \mathbb{Z}_{Q_5 P}^{DNUM}$$

$$\vec{g}^{-1}(a)[0] = [\,[a]_{D_0} \mid ////// \,]$$

modUp extension

$$\vec{g}^{-1}(a)[1] = [\, ////\ \mid [a]_{D_1} \mid ////// \,]$$

modUp extension

$$\vec{g}^{-1}(a)[2] = [\ 0 \quad 0 \quad 0\ ]$$

#6

full level

```
template<int N, int L, int DNUM>
void swkgen( const int Sfr[N],
             const int Sto[N],
             const uint64_t q[L],
             const uint64_t p[L/DNUM],
                   uint64_t swk[DNUM][2][L+(L/DNUM)][N] ){
    const int K = L/DNUM;
    uint64_t g[DNUM][L+K];
    gadget_g<L, DNUM>(q, p, g);
```

$\vec{g} \in \mathbb{Z}_{PQ}^{DNUM}$

```
    assert(L == K*DNUM);

    for(int n=0; n<DNUM; n++){
        uint64_t pt[L+K][N];
        for(int j=0; j<L; j++) {
```

pt = P·Sfr g[n]

mod(P, q[j])

```
            uint64_t P = 1;
            for(int k=0; k<K; k++)
                P = mul_mod(P, p[k]%q[j], q[j]);
```

pt = P·g[n]·Sfr

```
            uint64_t Pg = mul_mod(P, g[n][j], q[j]);
            for(int i=0; i<N; i++)
                pt[j][i] = mul_mod((q[j]+sfr[i])%q[j], Pg, q[j]);
        }
```

mod(pt, P)=0

```
        for(int j=0; j<K; j++)
            for(int i=0; i<N; i++)
                pt[j+L][i] = 0;
```

$\vec{\hat{swk}} = enc_{Sto}(P s_{fr} \vec{g})$

```
        uint64_t qp[L+K];
        for(int j=0; j<L; j++) qp[j] = q[j];
        for(int j=0; j<K; j++) qp[L+j] = p[j];

        enc<N, L+K>(pt, Sto, qp, swk[n]);
    }
}
```

```
template< int L, int DNUM, int K>
void gadget_g( const uint64_t q[L],
               const uint64_t p[K],
               uint64_t g[DNUM][L+K]){
  for(int d=0; d<DNUM; d++)
  for(int i=0; i<L+K ; i++)
     if((d*K <= i) && (i< (d+1)*K)&& (i<L))  g[d][i]= 1;
     else                                     g[d][i]=0;
}
```

L may not be the full level

$\vec{g} \in \mathbb{Z}_{QP}^{DNUM}$

When L=5, K=3, DNUM=3,

$\vec{g}$ = [11100000, 00011000, 00000000] : See page #5

```
template< int N, int L, int DNUM, int K>
void gadget_ginv( const uint64_t q[L],
                  const uint64_t p[K],
                  const uint64_t a[L][N],
                  uint64_t ginva[DNUM][L+K][N]){
 for(int d=0;d<DNUM;d++){
  uint64_t QP[L+K];
  for(int i=0; i<L; i++) QP[i]= q[i];
  for(int i=0; i<K; i++) QP[L+i]=p[i];

  int off1 = (d*K<L)? (d*K) : L;
  int off2 = ((d+1)*K<L)? ((d+1)*K): L;
  uint64_t temp[K];
  for(int i=off1; i<off2; i++) temp[i-off1]=QP[i];
  for(int i=off1-1; i>=0; i--) QP[i+off2-off1]=QP[i];
  for(int i=off1; i<off2; i++) QP[i-off1]=temp[i-off1];
  for(int i=off1; i<off2; i++)

  for(int i=off1; i<off2; i++)
  for(int j=0 ;j< N; j++)
     ginva[d][i-off1][j]= a[i][j];

  if(off2>off1)
     modUp<N, L+K>(QP, off2-off1, ginva[d]);

  for(int j=0; j<N; j++){
     for(int i=0; i<off2-off1; i++) temp[i]= ginva[d][i][j];
     for(int i=0; i<off1; i++) ginva[d][i][j]= ginva[d][i+off2-off1][j];
     for(int i=0; i<off2-off1; i++) ginva[d][i+off1][j]= temp[i];
  }
 }
}
```

$a \in R_Q$, $\vec{g}(a) \in R_{QP}^{DNUM}$

$QP = [q_0 q_1 q_2 | q_3 q_4 | p_0 p_1 p_2]$