# NIST 양자내성암호 공모전 KEM Finalist 최적 구현 동향

정보컴퓨터공학과 **권혁동**
IT융합공학부 엄시우 심민주 서화정†

한성대학교 HANSUNG UNIVERSITY
CryptoCraft LAB

서론

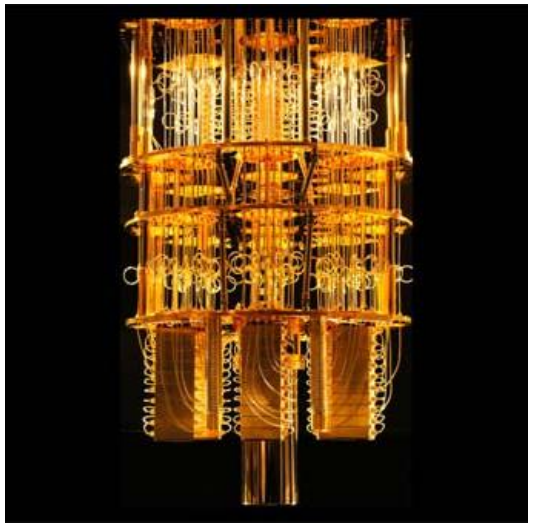
양자내성암호 공모전

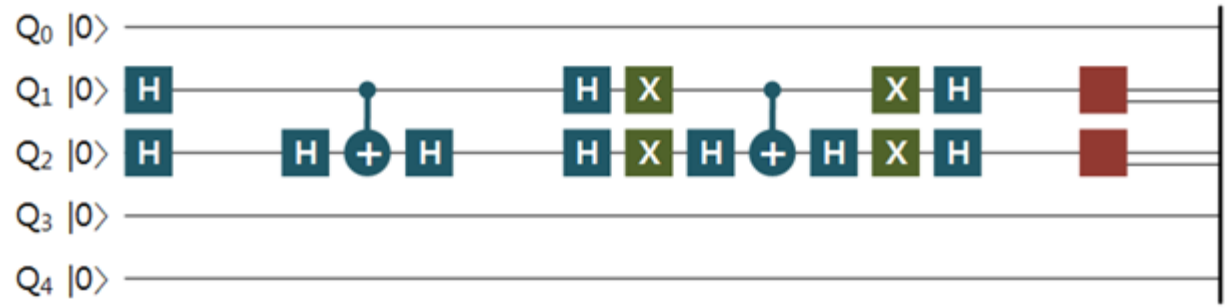
최적 구현 동향

결론

# 서론

- 양자컴퓨터는 **양자역학의 원리**를 사용하는 컴퓨터
  - 0과 1 전기 신호를 사용하는 고전컴퓨터에 비해 매우 빠름
- 양자컴퓨터 상에서는 특정 알고리즘의 실현이 가능
  - Grover 알고리즘: 검색에 능하며 Brute-force를 효과적으로 진행
  - Shor 알고리즘: 효과적인 소인수 분해 진행
- **기존 암호 체계의 붕괴를 야기**
- NIST의 양자내성암호 공모전 진행과 KEM 분야의 최적구현물을 확인





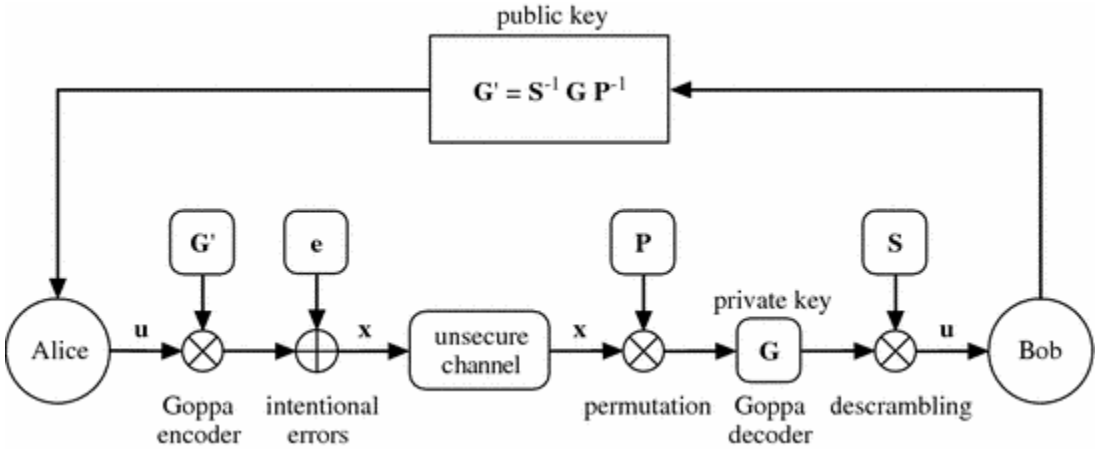
# 양자내성암호 공모전

- 2017년 NIST는 양자내성암호 표준화를 위한 공모전 개최
- 2020년 **Round 3 Final**까지 진행
  - 공개키 암호화 부문(KEM)
  - 전자서명 부문(Digital signature)
- KEM에서는 4종류의 암호 알고리즘이 진출
  - **Classic McEliece, CRYSTALS-KYBER, NTRU, Saber**

| | Finalists | Alternates |
|---|---|---|
| KEMs/Encryption | Kyber<br>NTRU<br>SABER<br>Classic McEliece | Bike<br>FrodoKEM<br>HQC<br>NTRUprime<br>SIKE |
| Signatures | Dilithium<br>Falcon<br>Rainbow | GeMSS<br>Picnic<br>SPHINCS+ |

# 최적 구현 동향: Classic McEliece

- 1979년 제안된 알고리즘으로 **부호 기반 알고리즘(Code based)**
  - Round 3 Finalist 중 가장 작은 암호문 크기 보유
  - 제안된 지 오래된 알고리즘으로 신뢰성과 안전성이 높은 특징
  - Finalist 중 유일한 부호 기반 알고리즘



# 최적 구현 동향: Classic McEliece

- ARM Cortex-M4를 대상으로 구현
  - STM32F4-Discovery Microcontroller
  - **Constant-time** 구현을 시도
  - RAM이 부족 (192KB) → **Flash memory (1MB)에 공개키 저장**
- 레벨 1 매개변수 세트 비교
  - FrodoKEM 레벨 1 매개변수 세트 대비 **Enc 80배, Dec 17배** 이상 빠름
  - 레벨 1 매개변수의 Enc에 한해서는 **다른 알고리즘과 대등한 연산 속도** 보유

| parameter set | level | decapsulation | encapsulation | key generation |
|---|---|---|---|---|
| mceliece348864f | 1 | 2 706 681 | 582 199 | 1 430 811 294 |
| mceliece348864 | 1 | | | 2 146 932 033 |
| mceliece460896* | 3 | 6 535 186 | 1 081 335 | |
| mceliece6688128* | 5 | 7 412 111 | | |
| mceliece8192128* | 5 | 7 481 747 | | |

| scheme (implementation) | level | key generation | encapsulation | decapsulation |
|---|---|---|---|---|
| frodokem640aes (m4) | 1 | 48 348 105 | 47 130 922 | 46 594 383 |
| kyber512 (m4) | 1 | 463 343 | 566 744 | 525 141 |
| kyber768 (m4) | 3 | 763 979 | 923 856 | 862 176 |
| lightsaber (m4f) | 1 | 361 687 | 513 581 | 498 590 |
| saber (m4f) | 3 | 654 407 | 862 856 | 835 122 |
| ntruhps2048509 (m4f) | 1 | 79 658 656 | 564 411 | 537 473 |
| ntruhps2048677 (m4f) | 3 | 143 734 184 | 821 524 | 815 516 |
| sikep434 (m4) | 1 | 48 264 129 | 78 911 465 | 84 276 911 |
| sikep610 (m4) | 3 | 119 480 622 | 219 632 058 | 221 029 700 |

# 최적 구현 동향: Classic McEliece

- Key Pair Generation을 중점적으로 구현한 구현물
  - Mceliece 8192128(레벨 5 매개변수)를 대상
  - **AVX(Advanced Vector eXtensions)**을 활용
  - Polynomial multiplication, Gaussian Reduction을 최적 구현 지점으로 설정
  - AVX를 사용하여 **구현에 필요한 명령어 숫자를 줄이는데 집중**
  - 기존 대비 **최대 55.57%**의 성능 향상

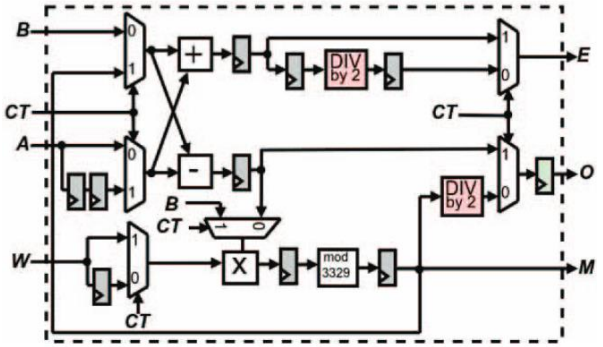
| Percentile | Original (ms) | Optimized (ms) | Speedup (%) |
|---|---|---|---|
| *Private key generation* | | | |
| 25% | 12, 201 | 4, 599 | 62.30 |
| 50% | 12, 356 | 4, 628 | 62.50 |
| 75% | 13, 421 | 4, 675 | 65.16 |
| *Public key generation* | | | |
| 25% | 111, 852 | 49, 388 | 55.84 |
| 50% | 115, 142 | 51, 385 | 55.37 |
| 75% | 127, 750 | 67, 702 | 47.00 |
| *Key-pair generation* | | | |
| 25% | 185, 268 | 98, 963 | 45.70 |
| 50% | 191, 184 | 107, 776 | 43.63 |
| 75% | 212, 542 | 118, 116 | 55.57 |

# 최적 구현 동향: CRYSTALS-KYBER

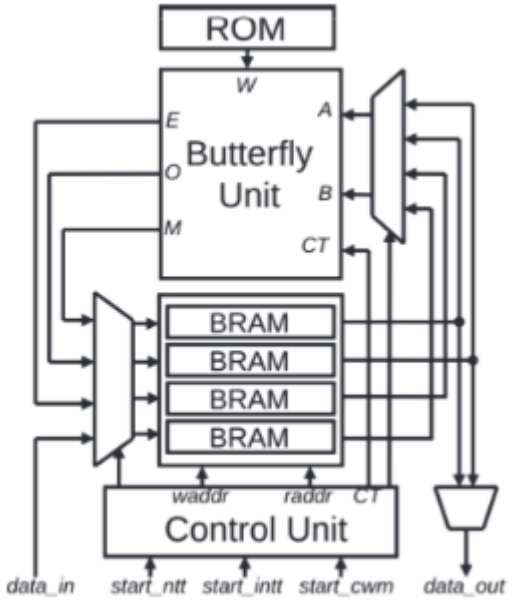
- Kyber는 전자서명 부문의 Dilithium과 쌍을 이루는 알고리즘
- Ring-LWE를 사용한 Module-LWE 기반의 알고리즘
  - KEM 부문 Finalist 중 **유일하게 LWE 사용**
  - Alternate를 포함할 경우, FrodoKEM 동일 기반 알고리즘
- Fujisaki-Okamoto 변환 사용
- Cyclotomic ring을 사용함
  - **NTT(Number Theoretic Transform)를 사용하여 효율적인 구현** 가능

# 최적 구현 동향: CRYSTALS-KYBER

- ARM Cortex series 상에서의 구현물
  - NTT, Inverse NTT, Polynomial multiplication 과정을 최적화
  - NTT를 최적화하기 위해 새로운 **Butterfly structure 구성**
  - 최대 16개의 unit을 사용
  - **NTT 112배, Inverse NTT 132배, Polynomial multiplication 109배 성능 향상**
- 제안하는 기법을 Keygen, Enc, Dec에 적용 가능




| Work | Platform | $n$ | $q$ / $\lceil \log_2(q) \rceil$ | LUT / REG / DSP / BRAM | Clock (MHz) | Latency (CC) | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | NTT | INTT | PMUL |
| [6][a] | Intel Corei7-6600U Skylake | 256 | 7681 / 13 | – / – / – / – | – | 419 | 394 | 1278 |
| | Intel Core i7-4770K Haswell | 256 | 7681 / 13 | – / – / – / – | – | 460 | 440 | 1432 |
| [7][a,b,c] | ARM Cortex-M4 | 256 | 7681 / 13 | – / – / – / – | – | 9452 | 10373 | 32576 |
| | | 256 | 3329 / 12 | – / – / – / – | – | 7725 | 9347 | 27873 |
| [10][a,b] | Zynq-7000 | 256 | – / 16 | 2908 / 170 / 9 / – | – | 1935 | 1930 | – |
| [11][a] | Virtex-6 | 256 | 7681 / 13 | 4549 / 3624 / 1 / 12 | 262 | 2096 | – | – |
| [12][a] | Virtex-6 | 256 | 7681 / 13 | 1349 / 860 / 1 / 2 | 313 | 1691 | – | – |
| | | 512 | 12289 / 14 | 1536 / 953 / 1 / 3 | 278 | 3443 | | |
| [13][b,c] | Artix-7, Virtex-7 | 256 | 3329 / 12 | – / – / – / – | 225 | 1834 | – | – |
| [14][a,b] | 28nm CMOS | 256 | 3329 / 12 | 512K / – / – / – | – | 41 | – | – |
| | | 256 | 7681 / 13 | | | 45 | | |
| TW-1 BTF[b,c] | Spartan-6 | 256 | 3329 / 12 | 985 / 444 / 1 / 5 | 138 | 904 | 904 | 3359 |
| | Artix-7 | | | 948 / 352 / 1 / 2.5 | 190 | | | |
| TW-4 BTFs[b,c] | Spartan-6 | | | 2498 / 1046 / 4 / 18 | 127 | 232 | 233 | 864 |
| | Artix-7 | | | 2543 / 792 / 4 / 9 | 182 | | | |
| TW-16 BTFs[b,c] | Spartan-6 | | | 9898 / 3688 / 16 / 70 | 115 | 69 | 71 | 256 |
| | Artix-7 | | | 9508 / 2684 / 16 / 35 | 172 | | | |



# 최적 구현 동향: CRYSTALS-KYBER

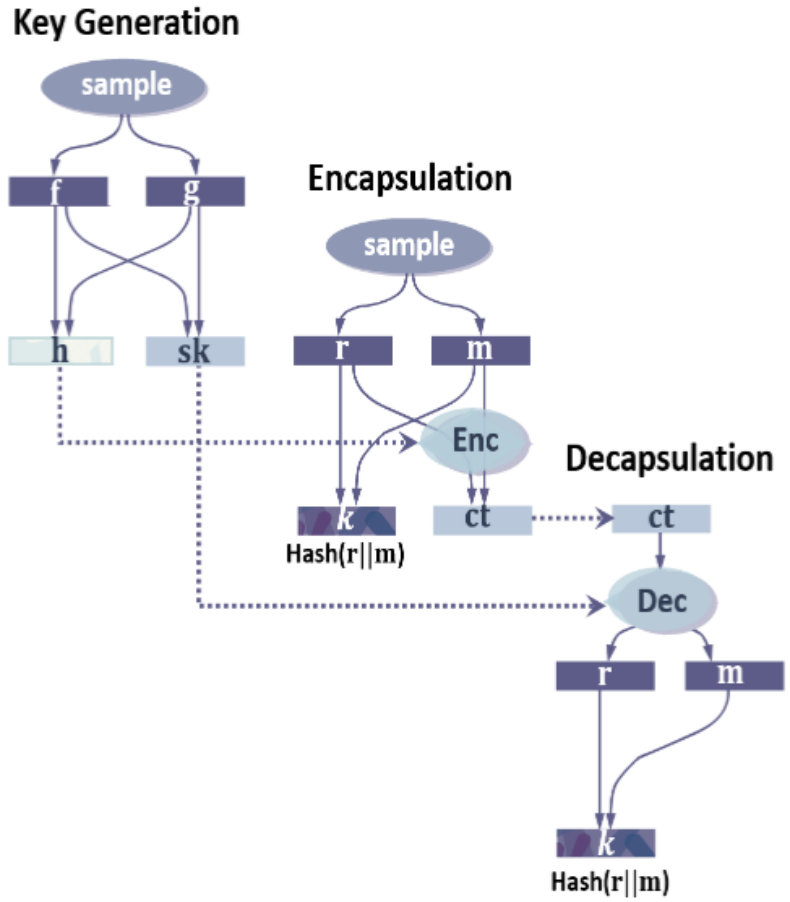
- ARM Cortex-A74 및 Apple M1 상에서의 구현
  - 모두 **ARMv8**에 속하는 프로세서로 동일한 기법을 적용
- **곱셈기를 최적 구현 지점**으로 사용
  - Barret multiplication, Montgomery multiplication
  - **병렬 연산자(NEON)을 사용**하여 구현에 필요한 명령어 수를 줄임
- 기존 대비 성능 향상을 확인 가능
  - 대체로 기존 대비 **1.3~1.5배의 성능 향상**

| Multiplication | Type | Instructions |
|---|---|---|
| Barret | Normal | 3 |
| Polynomial | Normal | 5 |
| | Constant | 4 |
| | Round off | 3 |
| | Big number | 7 |

| | Cortex-A72 | | | Apple M1 | | |
|---|---|---|---|---|---|---|
| | **K** | **E** | **D** | **K** | **E** | **D** |
| kyber512 (Ours) | 62 459 | 80 710 | 76 443 | 14 939 | 24 834 | 20 893 |
| kyber512 [NG21] | 67 903[a] | 88 906[a] | 87 563[a] | 23 000 | 32 500 | 29 400 |
| kyber512 [SKS+21][b] | 84 728 | 109 668 | 108 646 | – | – | – |
| kyber768 (Ours) | 99 201 | 127 453 | 120 665 | 23 756 | 36 284 | 31 047 |
| kyber768 [NG21] | 110 784[a] | 141 312[a] | 138 984[a] | 36 300 | 49 200 | 45 700 |
| kyber768 [SKS+21][b] | 143 791 | 180 687 | 179 085 | – | – | – |
| kyber1024 (Ours) | 156 694 | 192 280 | 184 161 | 33 024 | 48 925 | 44 000 |
| kyber1024 [NG21] | 176 809[a] | 215 665[a] | 214 076[a] | 55 900 | 71 600 | 67 100 |
| kyber1024 [SKS+21][b] | 228 082 | 272 418 | 270 668 | – | – | – |
| lightsaber (Ours) | 64 181 | 87 272 | 92 813 | 20 137 | 29 731 | 28 551 |
| lightsaber [NG21][c] | 83 960[a] | 118 583[a] | 136 203[a] | 31 200 | 37 200 | 35 300 |
| saber (Ours) | 109 192 | 140 103 | 147 925 | 32 865 | 44 917 | 44 074 |
| saber [NG21][c] | 158 757[a] | 206 337[a] | 226 304[a] | 51 300 | 59 900 | 58 000 |
| firesaber (Ours) | 175 104 | 211 382 | 222 317 | 50 345 | 65 402 | 64 593 |
| firesaber [NG21][c] | 245 249[a] | 304 128[a] | 330 750[a] | 77 000 | 87 900 | 86 700 |
| | **K** | **S** | **V** | **K** | **S** | **V** |
| dilithium2 (Ours) | 269 724 | 649 230 | 272 824 | 71 061 | 224 125 | 69 792 |
| dilithium2 (ref) | 410 312 | 1 353 753 | 449 633 | 187 842 | 741 140 | 199 615 |
| dilithium3 (Ours) | 515 776 | 1 089 387 | 447 460 | 152 435 | 365 248 | 104 821 |
| dilithium3 (ref) | 743 166 | 2 308 598 | 728 866 | 358 848 | 1 218 027 | 329 187 |
| dilithium5 (Ours) | 782 752 | 1 436 988 | 764 886 | 178 137 | 426 635 | 167 489 |
| dilithium5 (ref) | 1 151 504 | 2 903 604 | 1 198 723 | 544 833 | 1 531 067 | 557 696 |

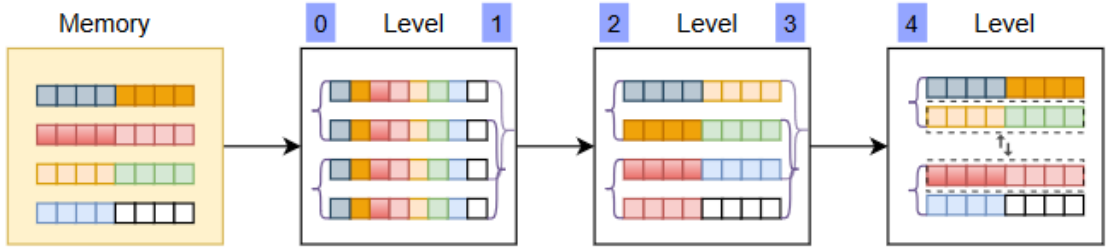
# 최적 구현 동향: NTRU

- 격자 기반 암호 알고리즘
  - Ring 구조를 활용
  - 초기 격자 기반 암호 알고리즘 중 하나
  - SVP(Shortest Vector Problem)을 사용
  - 공개키가 주어질 때 가장 짧은 벡터를 찾기 어려움
- **연산 속도가 빠르며 키 사이즈가 작음**



# 최적 구현 동향: NTRU

- ARM Cortex-A72와 Apple M1 상에서의 구현
  - ARMv8 프로세서군
  - Vector register를 사용한 병렬 구현
- NTT 연산을 최적 구현하는데 집중
  - 각 단계 별로 **연산에 필요한 값을 레지스터에 모아서 진행**
- 레퍼런스 구현물과 성능 비교 진행
  - HPS677: Enc는 **약 3.05~3.24배**, Dec는 **약 7.89~8.49배** 성능 향상
  - HRSS701: Enc는 **약 6.68배**, Dec는 **약 7.24배** 성능 향상
  - AVX 구현물에 비해서는 떨어지는 성능




| Algorithm | ref (kc) | | neon (kc) | | AVX2 (kc) | | ref/neon | | AVX2/neon | |
|---|---|---|---|---|---|---|---|---|---|---|
| | E | D | E | D | E | D | E | D | E | D |
| lightsaber | 50.9 | 54.9 | 37.2 | 35.3 | 41.9 | 42.2 | 1.37 | 1.55 | 1.13 | 1.19 |
| kyber512 | 75.7 | 89.5 | 32.6 | 29.4 | 28.4 | 22.6 | 2.33 | 3.04 | 0.87 | 0.77 |
| ntru-hps677 | 183.1 | 430.4 | 60.1 | 54.6 | 26.0 | 45.7 | 3.05 | 7.89 | 0.43 | 0.84 |
| ntru-hrss701 | 152.4 | 439.9 | 22.8 | 60.8 | 20.4 | 47.7 | 6.68 | 7.24 | 0.90 | 0.78 |
| saber | 90.4 | 96.2 | 59.9 | 58.0 | 70.9 | 70.7 | 1.51 | 1.66 | 1.18 | 1.22 |
| kyber768 | 119.8 | 137.8 | 49.2 | 45.7 | 43.4 | 35.2 | 2.43 | 3.02 | 0.88 | 0.77 |
| ntru-hps821 | 245.3 | 586.5 | 75.7 | 69.0 | 29.9 | 57.3 | 3.24 | 8.49 | 0.39 | 0.83 |
| firesaber | 140.9 | 150.8 | 87.9 | 86.7 | 103.3 | 103.7 | 1.60 | 1.74 | 1.18 | 1.20 |
| kyber1024 | 175.4 | 198.4 | 71.6 | 67.1 | 63.0 | 53.1 | 2.45 | 2.96 | 0.88 | 0.79 |

# 최적 구현 동향: Saber

- Module-LWR 방식의 격자 기반 암호
- ARM Cortex series 상에서 구현
  - **Kyber와 동일한 최적 구현 기법 적용**
  - 단, **32-bit용 NTT** 구현 및 연산자를 사용
  - LightSaber: 약 1.23~1.54배 성능 향상
  - Saber: 약 1.31~1.55배 성능 향상
  - FireSaber: 약 1.34~1.52배 성능 향상

| | Cortex-A72 | | | Apple M1 | | |
|---|---|---|---|---|---|---|
| | **K** | **E** | **D** | **K** | **E** | **D** |
| kyber512 (Ours) | 62 459 | 80 710 | 76 443 | 14 939 | 24 834 | 20 893 |
| kyber512 [NG21] | 67 903[a] | 88 906[a] | 87 563[a] | 23 000 | 32 500 | 29 400 |
| kyber512 [SKS+21][b] | 84 728 | 109 668 | 108 646 | – | – | – |
| kyber768 (Ours) | 99 201 | 127 453 | 120 665 | 23 756 | 36 284 | 31 047 |
| kyber768 [NG21] | 110 784[a] | 141 312[a] | 138 984[a] | 36 300 | 49 200 | 45 700 |
| kyber768 [SKS+21][b] | 143 791 | 180 687 | 179 085 | – | – | – |
| kyber1024 (Ours) | 156 694 | 192 280 | 184 161 | 33 024 | 48 925 | 44 000 |
| kyber1024 [NG21] | 176 809[a] | 215 665[a] | 214 076[a] | 55 900 | 71 600 | 67 100 |
| kyber1024 [SKS+21][b] | 228 082 | 272 418 | 270 668 | – | – | – |
| lightsaber (Ours) | 64 181 | 87 272 | 92 813 | 20 137 | 29 731 | 28 551 |
| lightsaber [NG21][c] | 83 960[a] | 118 583[a] | 136 203[a] | 31 200 | 37 200 | 35 300 |
| saber (Ours) | 109 192 | 140 103 | 147 925 | 32 865 | 44 917 | 44 074 |
| saber [NG21][c] | 158 757[a] | 206 337[a] | 226 304[a] | 51 300 | 59 900 | 58 000 |
| firesaber (Ours) | 175 104 | 211 382 | 222 317 | 50 345 | 65 402 | 64 593 |
| firesaber [NG21][c] | 245 249[a] | 304 128[a] | 330 750[a] | 77 000 | 87 900 | 86 700 |
| | **K** | **S** | **V** | **K** | **S** | **V** |
| dilithium2 (Ours) | 269 724 | 649 230 | 272 824 | 71 061 | 224 125 | 69 792 |
| dilithium2 (ref) | 410 312 | 1 353 753 | 449 633 | 187 842 | 741 140 | 199 615 |
| dilithium3 (Ours) | 515 776 | 1 089 387 | 447 460 | 152 435 | 365 248 | 104 821 |
| dilithium3 (ref) | 743 166 | 2 308 598 | 728 866 | 358 848 | 1 218 027 | 329 187 |
| dilithium5 (Ours) | 782 752 | 1 436 988 | 764 886 | 178 137 | 426 635 | 167 489 |
| dilithium5 (ref) | 1 151 504 | 2 903 604 | 1 198 723 | 544 833 | 1 531 067 | 557 696 |

# 결론

- NIST 양자내성암호 공모전 Round 3 Finalist KEM 부문 알고리즘 확인
- 다양한 플랫폼 상에서 최적 구현 조사
  - 많은 구현물이 **ARM 프로세서 상**에서 구현을 진행
  - 강력한 **병렬 구현 연산자 및 환경** 지원
  - **하드웨어 구현**의 경우 굉장히 뛰어난 연산 성능 향상
- 양자내성암호는 연산 성능이 떨어지는 경우가 존재
  - 원활한 보급 및 사용을 위해서는 최적 구현의 연구가 중요
  - 지속적인 연구가 필요할 것으로 전망됨

# Q & A